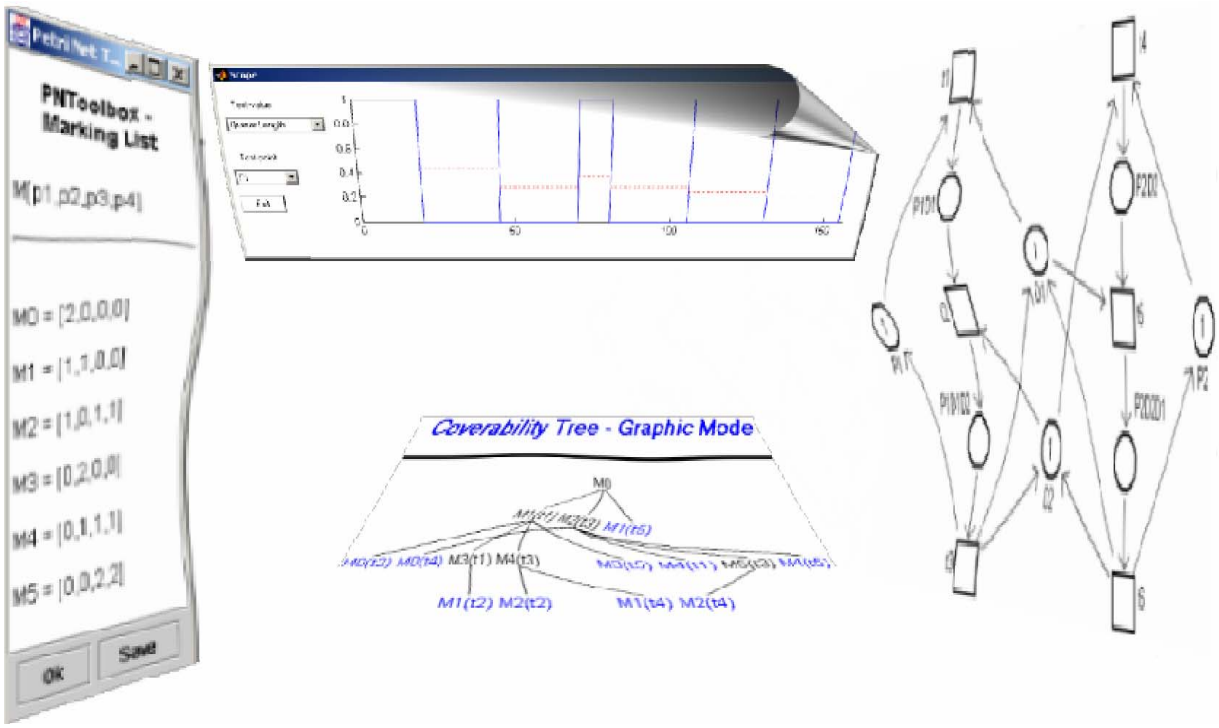
OCTAVIAN PĂSTRĂVANU

MIHAELA MATCOVSCHI CRISTIAN MAHULEA

APLICAȚII ALE REȚELELOR PETRI ÎN STUDIEREA SISTEMELOR CU EVENIMENTE DISCRETE



Editura Gh. ASACHI

2002

OCTAVIAN PĂSTRĂVANU

MIHAELA MATCOVSCHI CRISTIAN MAHULEA

APLICAŢII ALE REŢELELOR PETRI ÎN STUDIEREA SISTEMELOR CU EVENIMENTE DISCRETE

Editura Gh. ASACHI

2002

Editura „GH. ASACHI”
Universitatea Tehnică Iaşi
Bd. D. Mangeron nr. 67
6600, Iaşi, Romania
Tel: 0232 – 23.13.43
Fax: 0232 – 21.42.90

Director editură:
Prof. univ. dr. ing. Mihail VOICU

Redactor:
Prof. Georgeta ANICULĂESEI

Referenți ştiințifici:
Prof. univ. dr. ing. Mihail Voicu
Prof. univ. dr. ing. Doru Pănescu

Colecția: Automatică şi informatică industrială

Descrierea CIP a Bibliotecii Naționale a României
PĂSTRĂVANU, OCTAVIAN
Aplicații ale rețelelor Petri în studierea sistemelor cu evenimente discrete / Octavian Păstrăvanu, Mihaela Matcovschi, Cristian Mahulea – Iaşi: Editura Gheorghe Asachi, 2002.
p.: cm. 16,5 x 22,5
ISBN: 973-8292-86-7

I. Matcovschi, Mihaela
II. Mahulea, Cristian

517.938
681.5

© Octavian PĂSTRĂVANU, Mihaela MATCOVSCHI, Cristian MAHULEA
Universitatea Tehnică “Gh. Asachi” Iaşi
Facultatea de Automatică şi Calculatoare
Bd. D. Mangeron 53A, 6600 Iaşi
email: opastrav@delta.ac.tuiasi.ro

Prefață

Ideea de a elabora lucrarea de față s-a conturat pe fondul evoluției, într-un ritm deosebit de alert, a domeniului sistemelor cu evenimente discrete, din dorința de a oferi un material care să furnizeze atât informații teoretice ample, cât și o bogată ilustrare a utilizării lor în practică. Deși, prin conținutul transmis, cartea posedă un caracter monografic, prin modul de structurare vizează, drept principal obiectiv, construirea raționamentelor științifice specifice domeniului, manipulând cunoștințele în contexte aplicative variate și relevante. Fiecare capitol al cărții conține un substanțial breviar teoretic, oferind cititorului toate noțiunile și rezultatele de bază care, ulterior, sunt angrenate în rezolvarea de probleme, soluțiile fiind analitice și/sau asistate de calculator. O contribuție deosebită la dezvoltarea abilităților de investigare este adusă de o serie de aplicații ce compară și corelează rezultatele rezolvărilor analitice și, respectiv, asistate de calculator.

În textele breviarelor teoretice, numerotarea relațiilor și figurilor este realizată cu un set de trei numere, precedat de abrevierea BT (Breviar Teoretic); primul număr coincide cu numărul capitolului, cel de al doilea este numărul secțiunii, iar cel de al treilea crește în secvență naturală. Similar, în textele aplicațiilor (enunțuri și soluții), numerotarea relațiilor și figurilor este realizată cu un set de trei numere, precedat de abrevierea AP (Aplicații); primul număr coincide cu numărul capitolului, cel de al doilea este numărul aplicației, iar cel de al treilea crește în secvență naturală.

În scopul abordării asistate de calculator a rezolvărilor a fost utilizat mediul Petri Net Toolbox, dezvoltat de autorii cărții pentru a rula sub MATLAB. Facilitățile acestuia sunt descrise în ultimul capitol, care reproduce forma originală, în limba engleză, a informațiilor din Help-ul respectivului software.

Întregul material a fost de așa manieră conceput încât să asigure adresabilitate pentru o sferă largă de cititori cu formație tehnico-științifică, incluzând specialiști, cercetători, cadre didactice, studenți și doctoranzi. Stilul de prezentare se concretizează într-o expunere autonomă, ce nu necesită alte lecturi în paralel, dar, totodată, cititorului preocupat de aprofundarea anumitor teme îi sunt indicate referințe bibliografice

generoase. Modul de organizare al lucrării nu obligă la parcurgerea integrală a textului, permiţând o lectură selectivă, orientată în conformitate cu nivelul de pregătire şi problematica de interes a fiecărui cititor în parte, ghidarea realizându-se cu multă uşurinţă pe seama cuprinsului anume elaborat cu un grad relevant de detaliere. Pe lângă acest cuprins, o imagine generală despre modul de integrare a capitolelor cărţii în economia de ansamblu rezultă din parcurgerea Capitolului 1.

În particular, lucrarea de faţă (în totalitate, ori anumite porţiuni ale ei) poate servi şi drept suport pentru cursuri destinate studenţilor sau pentru cursuri de perfecţionare destinate absolvenţilor, stilul de prezentare bazându-se pe principiile instruirii graduale, cu acordarea unei ponderi importante exemplelor, studiilor practice şi utilizării facilităţilor software.

Pentru a reproduce soluţiile asistate de calculator din lucrare, sau pentru a dezvolta noi aplicaţii sub Petri Net Toolbox, cititorii se pot adresa autorilor cărţii spre a intra în posesia unei versiuni de testare a acestui pachet, în regim gratuit.

Rod al unei colaborări de lungă durată dintre trei cadre didactice de la Catedra de automatică şi informatică industrială a Universităţii Tehnice "Gh. Asachi" din Iaşi, prezenta lucrare pune în valoare atât viziunea proprie a autorilor, creată în urma consultării unui vast material bibliografic, cât şi experienţa dobândită de aceştia în utilizarea reţelelor Petri pentru modelarea, analiza şi proiectarea sistemelor cu evenimente discrete. Cartea reflectă totodată şi preocupările educaţionale ale celor trei autori care au pregătit cursuri şi şedinţe de aplicaţii dedicate disciplinei Sisteme dinamice cu evenimente discrete existentă din anul 1996 în planurile de învăţământ ale Facultăţii de Automatică şi Calculatoare din Iaşi.

În încheiere, autorii doresc să îşi exprime gratitudinea faţă de cei doi referenţi, ale căror personalităţi academice marcante au girat apariţia acestei cărţi, precum şi faţă de colectivul Editurii "Gh. Asachi" care a inclus volumul în planul său editorial.

Publicarea cărţii s-a bucurat de suportul financiar al Consiliului Naţional pentru Finaţarea Învăţamântului Superior, prin intermediul proiectului LICAP ("Laborator de instruire în domeniul conducerii asistate de calculator a proceselor").

Mulţumiri anticipate sunt adresate tuturor cititorilor, care, prin observaţiile ce le vor formula în urma consultării acestei cărţi, vor contribui la îndepărtarea unor posibile neajunsuri ale textului şi la îmbunătăţirea calităţii.

Noiembrie 2002 *Autorii*

Cuprins

Prefaţa i
Cuprins iii
Abstract and Contents vii
Simboluri şi notaţii xi

Cap. 1. *Introducere* 1

Cap. 2. *Modele de tip reţea Petri netemporizată* 7
Breviar teoretic 7
BT2.1. Conceptul de reţea Petri netemporizată 7
BT2.2. Terminologie uzuală 8
BT2.3. Validarea şi executarea tranziţiilor în reţele cu capacitate infinită – evoluţia stărilor 9
BT2.4. Unele extensii pentru reţele Petri netemporizate 10
BT2.4.1. Reţele Petri cu capacitate finită 10
BT2.4.2. Reţele cu probabilităţi şi priorităţi 10
BT2.4.3. Reţele cu arce inhibitoare 11
BT2.5. Modelarea cu reţele Petri ordinare 11
BT2.5.1. Structuri tipice utilizate în modelare 11
BT2.5.2. Capacitatea de modelare a unor subclase de reţele Petri ordinare 12
Aplicaţii (**AP2.1** – **AP2.5**) 14

Cap. 3. *Studierea proprietăţilor comportamentale* 27
Breviar teoretic 27
BT3.1. Definirea proprietăţilor comportamentale 27
BT3.1.1. Accesibilitate 27
BT3.1.2. Mărginire 27
BT3.1.3. Viabilitate 28
BT3.1.4. Reversibilitate 28
BT3.2. Producerea fenomenului de deadlock în sistemele cu resurse partajate 28
BT3.3.Arbori şi grafuri de acoperire/accesibilitate 29
BT3.3.1. Arbori de acoperire/accesibilitate 29
BT3.3.2. Grafuri de acoperire/accesibilitate 30

BT3.4. Ecuaţia de stare ... 30
BT3.5. Proprietăţi ce decurg din apartenenţa reţelelor la anumite subclase de reţele Petri ordinare ... 32
BT3.5.1. Criterii de viabilitate şi siguranţă a maşinilor de stare ... 33
BT3.5.2. Criterii de viabilitate şi siguranţă a grafurilor marcate ... 33
Aplicaţii (**AP3.1** – **AP3.6**) ... 34

Cap. 4. *Controlul procedural al sistemelor cu evenimente discrete* ... 49
Breviar teoretic ... 49
BT4.1. Specificaţii de proiectare pentru structurile de conducere ... 49
BT4.2. Proiectarea structurilor de conducere prin tehnici de sinteză hibridă ... 50
BT4.2.1. Rafinarea operaţiilor prin sinteză descendentă ... 50
BT4.2.1.1. Prezentarea generală a procedurii de rafinare ... 50
BT4.2.1.2. Module standard utilizate în rafinare ... 52
BT4.2.2. Ataşarea resurselor prin sinteză ascendentă ... 55
BT4.2.2.1. Prezentarea generală a procedurii de ataşare a resurselor ... 55
BT4.2.2.2. Detalierea Pasului 1 (ataşarea resurselor specifice nepartajate) ... 55
BT4.2.2.3. Detalierea Pasului 2 (ataşarea resurselor de stocare) ... 56
BT4.2.2.4. Detalierea Pasului 3 (ataşarea resurselor specifice partajate) ... 58
BT4.3. Proprietăţi caracteristice ale structurilor de conducere rezultate din sinteză ... 65
BT4.4. Funcţiuni de bază şi consideraţii de proiectare ale controlerului procedural ... 66
BT4.4.1. Funcţiuni de bază ... 66
BT4.4.2. Consideraţii de proiectare ... 66
Aplicaţii (**AP4.1** – **AP4.3**) ... 67

Cap. 5. *Studierea proprietăţilor structurale* ... 85
Breviar teoretic ... 85
BT5.1. Definirea proprietăţilor structurale şi criterii de caracterizare ... 85
BT5.1.1. Mărginire structurală ... 85
BT5.1.2. Conservativitate ... 86
BT5.1.3. Repetitivitate ... 86
BT5.1.4. Consistenţă ... 87
BT5.1.5. Relaţii între proprietăţile structurale ... 87
BT5.2. Invarianţi ... 88
BT5.2.1. Definiţia invarianţilor ... 88
BT5.2.2. Invarianţi în reţele duale şi/sau inversate ... 89
BT5.2.3. Criterii de caracterizare a invarianţilor şi conexiuni cu proprietăţile structurale ... 90
Aplicaţii (**AP5.1** – **AP5.4**) ... 91

Cap. 6. *Modele de tip reţea Petri cu temporizare deterministă* ... 103
Breviar teoretic ... 103
BT6.1. Reţele cu tranziţii temporizate ... 103
BT6.2. Reţele cu poziţii temporizate ... 104

BT6.3. Transformarea unei rețele temporizate *T* într-o rețea temporizată *P* 105
BT6.4. Transformarea unei rețele temporizate *P* într-o rețea temporizată *T* 106
BT6.5. Comportarea periodică a rețelelor Petri acoperite de invarianți *P*, (parțial) consistente, temporizate *T* 107
BT6.6. Comportarea periodică a rețelelor Petri acoperite de invarianți *P*, (parțial) consistente, temporizate *P* 109
Aplicații (**AP6.1** – **AP6.8**) 110

Cap. 7. *Modele de tip rețea Petri cu temporizare stohastică* 129
Breviar teoretic 129
BT7.1. Principiile temporizării stohastice 129
BT7.2. Variabile aleatoare - scurtă trecere în revistă 130
BT7.2.1. Variabile aleatoare 130
BT7.2.2. Funcție de repartiție. Densitate de repartiție 130
BT7.2.3. Funcții de variabile aleatoare 132
BT7.2.4. Caracteristici ale distribuțiilor de probabilitate 132
BT7.2.4.1. Valoare medie 132
BT7.2.4.2. Dispersie (varianță) 133
BT7.2.4.3. Moment de ordin p 134
BT7.2.4.4. Covarianță 134
BT7.2.5. Legea numerelor mari. Legi limită 135
BT7.2.6. Distribuții frecvent folosite în temporizarea stohastică 137
BT7.3. Procese stohastice. 137
BT7.4. Modele de tip rețea Petri cu temporizare stohastică 138
Aplicații (**AP7.1** – **AP7.5**) 138

Cap. 8. *Modele de tip rețea Petri stohastică* 153
Breviar teoretic 153
BT8.1. Rețele Petri stohastice 153
BT8.2. Proprietăți ale legii de distribuție exponențială 154
BT8.3. Lanțuri Markov omogene, continue în timp 155
BT8.3.1. Concepte de bază 155
BT8.3.2. Analiza tranzițiilor de stare a unui lanț Markov continuu şi omogen 157
BT8.4. Proprietăți ale rețelelor Petri stohastice 157
BT8.4.1. Construcția lanțului Markov asociat unei rețele Petri stohastice 157
BT8.4.2. Evaluarea performanțelor unei rețele Petri stohastice 158
BT8.4.3. Proprietăți de conservare 159
BT8.5. Rețele Petri stohastice generalizate 160
Aplicații (**AP8.1** – **AP8.4**) 161

Cap. 9. *Modele de tip max-plus* 175
Breviar teoretic 175
BT9.1. Proprietăți fundamentale ale algebrei (max, +) 175
BT9.2 Operații cu matrice şi vectori definiți pe algebra $(\mathbb{R} \cup \{-\infty\}, \max, +)$ 176

BT9.3 Reprezentări de stare în algebra max-plus pentru rețele Petri de tip graf marcat, temporizate *P* 178
BT9.4. Rezolvarea ecuației de stare 181
BT9.5. Rezolvarea ecuației de ieşire 185
Aplicații (**AP9.1** – **AP9.4**) 186

Cap. 10. *Petri Net Toolbox – descriere şi utilizare / Learning about Petri Net Toolbox* 201
Introduction – Petri Net Toolbox at a First Glance 201
Part I. Describing the Graphical User Interface (GUI) 204
I.1. Overview 204
I.2. Menu Bar 204
I.3. Quick Access Toolbar 208
I.4. Drawing Area 208
I.5. Drawing Panel 208
I.6. Draw/Explore Switch 208
I.7. Simulation Panel 209
I.8. Status Panel 209
I.9. Message Box 209
Part II. Exploiting the Toolbox 210
II.1. Building a Model 210
II.2. Exploring Properties 216
II.3. Running a Simulation 217
II.4. Analyzing Simulation Results 221
II.5. Max-Plus Models 222
II.6. Design 223
Appendix 1: XML file-format of a file containing a PN model 225
Appendix 2: Configuration File for the Petri Net Toolbox 229

Bibliografie 231
Index şi dicționar român-englez 235

APPLICATIONS OF PETRI NETS IN STUDYING DISCRETE EVENT SYSTEMS

Abstract and Contents

The book constructs a theoretical and practical framework, based on Petri net models, for exploring event-driven dynamics. This framework encompasses both untimed and timed Petri nets, the usage of the key concepts and results being illustrated by problems with analytical or computer-aided solutions. For computational approaches, the software environment ***Petri Net Toolbox***, developed by the authors to ensure full compatibility with MATLAB, has been used. The last chapter creates an overview of the capabilities and exploitation of this software. The book is self-contained, but the bibliographic list suggests numerous works recommended for getting a deeper insight into this field. The book was designed to answer the instructional needs for a large area of potential beneficiaries, namely people interested in a gradual and methodological training in discrete-event systems, which obviously includes undergraduate, postgraduate and doctoral students, as well as research workers and practitioners.

Preface i
Contents (in Romanian) iii
Abstract and Contents (in English) vii
Symbols and Notations xi

Chapter 1. *Introduction* 1

Chapter 2. *Untimed Petri Net Models* 7
Theory 7
BT2.1. The concept of untimed Petri net 7
BT2.2. Terminology and basic concepts 8
BT2.3. Transition enabling and firing in infinite-capacity Petri nets – State evolution 9
BT2.4. Extensions of untimed Petri nets 10
BT2.4.1. Finite-capacity Petri nets 10
BT2.4.2. Nets with probabilities and priorities 10
BT2.4.3. Nets with inhibitor arcs 11
BT2.5. Modeling with ordinary Petri nets 11
BT2.5.1. Typical structures used in modeling with Petri nets 11
BT2.5.2. Some subclasses of ordinary Petri nets and their modeling power 12
Applications (**AP2.1** – **AP2.5**) 14

Chapter 3. *Analysis Techniques for Behavioral Properties* 27
Theory 27
BT3.1. Definitions of the behavioral properties 27
BT3.1.1. Reachability 27
BT3.1.2. Boundedness 27
BT3.1.3. Liveness 28
BT3.1.4. Reversibility 28
BT3.2. Deadlock in systems with shared resources 28
BT3.3. Coverability/accessibility trees and graphs 29
BT3.3.1. Coverability/accessibility trees 29
BT3.3.2. Coverability/accessibility graphs 30
BT3.4. State equation 30
BT3.5. Behavioral properties induced by membership in some subclasses of ordinary Petri nets 32
BT3.5.1. Criteria for liveness and safeness of state machines 33
BT3.5.2. Criteria for liveness and safeness of marked graphs 33
Applications (**AP3.1** – **AP3.6**) 34

Chapter 4. *Procedural Control of Discrete Event Systems* 49
Theory 49
BT4.1. Design specifications for the control structure 49
BT4.2. Hybrid synthesis techniques 50
BT4.2.1. Refinement of the operation places by top–down synthesis 50
BT4.2.1.1. Overview of the procedure 50
BT4.2.1.2. Standard modules used in the refinement 52
BT4.2.2. Addition of the resource places by bottom-up synthesis 55
BT4.2.2.1. Overview of the procedure 55
BT4.2.2.2. Detailing Step 1 (addition of the nonshared resource places) 55
BT4.2.2.3. Detailing Step 2 (addition of the buffer places) 56
BT4.2.2.4. Detailing Step 3 (addition of the shared resource places) 58
BT4.3. Characteristic properties of the control structures resulted from synthesis 65
BT4.4. Basic actions and design issues of the procedural controller 66
BT4.4.1. Basic actions 66
BT4.4.2. Design issues 66
Applications (**AP4.1** – **AP4.3**) 67

Chapter 5. *Analysis Techniques for Structural Properties* 85
Theory 85
BT5.1. Definitions of the structural properties and criteria for their characterization 85
BT5.1.1. Structural liveness 85
BT5.1.2. Conservativeness 86
BT5.1.3. Repetitiveness 86
BT5.1.4. Consistency 87
BT5.1.5. Relationships between the structural properties 87

BT5.2. Invariants ... 88
BT5.2.1. Definition of the invariants ... 88
BT5.2.2. Invariants in reverse-dual nets ... 89
BT5.2.3. Criteria for the characterization of invariants and their relationships to structural properties ... 90
Applications (**AP5.1 – AP5.4**) ... 91

Chapter 6. *Deterministic Timed Petri Net Models* ... 103
Theory ... 103
BT6.1. Transition-timed nets ... 103
BT6.2. Place-timed nets ... 104
BT6.3. Transforming a *T*-timed net into a *P*-timed net ... 105
BT6.4. Transforming a *P*-timed net into a *T*-timed net ... 106
BT6.5. Periodic behavior of *T*-timed, (partially) consistent Petri nets, covered by *P*-invariants ... 107
BT6.6. Periodic behavior of *P*-timed, (partially) consistent Petri nets, covered by *P*-invariants ... 109
Applications (**AP6.1 – AP6.8**) ... 110

Chapter 7. *Stochastic Timed Petri Net Models* ... 129
Theory ... 129
BT7.1. Principles of stochastic timing ... 129
BT7.2. Random variables – overview ... 130
BT7.2.1. Random variables ... 130
BT7.2.2. Cumulative distribution function. Probability density function ... 130
BT7.2.3. Functions of random variables ... 132
BT7.2.4. Characteristics of probability distributions ... 132
BT7.2.4.1. Mean value (expectation) ... 132
BT7.2.4.2. Variance ... 133
BT7.2.4.3. *k*-th order moment ... 134
BT7.2.4.4. Covariance ... 134
BT7.2.5. Law of large numbers. Limit Theorems ... 135
BT7.2.6. Probability distributions frequently used in stochastic timing ... 137
BT7.3. Stochastic processes. ... 137
BT7.4. Stochastic timed Petri nets ... 138
Applications (**AP7.1 – AP7.5**) ... 138

Chapter 8. *Stochastic Petri Net Models* ... 153
Theory ... 153
BT8.1. Stochastic Petri Nets ... 153
BT8.2. Properties of the exponential distribution ... 154
BT8.3. Continuous time homogenous Markov chains ... 155
BT8.3.1. Basic concepts ... 155

BT8.3.2. Analysis of state transitions for continuous time homogenous Markov chains ... 157
BT8.4. Properties of stochastic Petri nets ... 157
BT8.4.1. Construction of the Markov chain associated with a stochastic Petri net ... 157
BT8.4.2. Performance evaluation of a stochastic Petri net ... 158
BT8.4.3. Conservativeness properties ... 159
BT8.5. Generalized stochastic Petri nets ... 160
Applications (**AP8.1** – **AP8.4**) ... 161

Chapter 9. *Max-plus Models* ... 175
Theory ... 175
BT9.1. Basic properties of the (max, +) algebra ... 175
BT9.2. Operations with vectors and matrices over the $(\mathbb{R} \cup \{-\infty\}, \max, +)$ algebra ... 176
BT9.3. State-space representations over the max-plus algebra for *P*-timed marked graphs ... 178
BT9.4. Solving the state equation ... 181
BT9.5. Solving the output equation ... 185
Applications (**AP9.1** – **AP9.4**) ... 186

Chapter 10. *Petri Net Toolbox – overview and utilization / Learning about Petri Net Toolbox* ... 201
Introduction – Petri Net Toolbox at a First Glance ... 201
Part I. Describing the Graphical User Interface (GUI) ... 204
I.1. Overview ... 204
I.2. Menu Bar ... 204
I.3. Quick Access Toolbar ... 208
I.4. Drawing Area ... 208
I.5. Drawing Panel ... 208
I.6. Draw/Explore Switch ... 208
I.7. Simulation Panel ... 209
I.8. Status Panel ... 209
I.9. Message Box ... 209
Part II. Exploiting the Toolbox ... 210
II.1. Building a Model ... 210
II.2. Exploring Properties ... 216
II.3. Running a Simulation ... 217
II.4. Analyzing Simulation Results ... 221
II.5. Max-Plus Models ... 222
II.6. Design ... 223
Appendix 1: XML file-format of a file containing a PN model ... 225
Appendix 2: Configuration File for the Petri Net Toolbox ... 229

Bibliography ... 231
Index and Romanian-English Dictionary ... 235

Simboluri şi notaţii

$\mathbb{R}$ – mulţimea numerelor reale

$\mathbb{N}$ – mulţimea numerelor naturale (inclusiv 0)

$\mathbb{Z}$ – mulţimea numerelor întregi

$\in$ – apartenenţă

$\subset$ – incluziune strictă

$\subseteq$ – incluziune cu posibilitate de egal

$*^T$ – transpusa matricei $*$

rang$*$ – rangul matricei $*$

$|*|$ – cardinalitatea mulţimii $*$

$\boldsymbol{x} < \boldsymbol{y}$ – inegalităţi $<$ pe toate componentele vectorilor $\boldsymbol{x}$, $\boldsymbol{y}$

$\boldsymbol{x} \leq \boldsymbol{y}$ – inegalităţi $\leq$ pe toate componentele vectorilor $\boldsymbol{x}$, $\boldsymbol{y}$

$\boldsymbol{x} <\neq \boldsymbol{y}$ – inegalităţi $\leq$ pe toate componentele vectorilor $\boldsymbol{x}$, $\boldsymbol{y}$, cu cel puţin o inegalitate strictă ($<$)

$\boldsymbol{x} >\neq \boldsymbol{y}$ – inegalităţi $\geq$ pe toate componentele vectorilor $\boldsymbol{x}$, $\boldsymbol{y}$, cu cel puţin o inegalitate strictă ($>$)

$P = \{p_1, ..., p_m\}$ – mulţimea poziţiilor unei reţele Petri

$T = \{t_1, ..., t_n\}$ – mulţimea tranziţiilor unei reţele Petri

$W(t_i, p_j)$ – ponderea arcului de la t_i la p_j

$W(p_j, t_i)$ – ponderea arcului de la p_j la t_i

$M \in \mathbb{Z}^m$ – vectorul de marcaj

$M(p_j) \in \mathbb{N}$ – marcajul poziţiei p_j

$M_0 \in \mathbb{Z}^m$ – vectorul marcajului iniţial

$M_0(p_j) \in \mathbb{Z}$ – marcajul iniţial al poziţiei p_j

$\boldsymbol{N}$ – topologia unei reţele Petri

$(\boldsymbol{N}, M_0)$ – topologia unei reţele Petri cu un marcaj iniţial dat

<*> – suportul invariantului $*$ (de tip P sau T) al unei reţele Petri

$^\bullet *$ – mulţimea predecesor (mulţime de tranziţii, respectiv poziţii) a mulţimii $*$ (mulţime de poziţii, respectiv tranziţii)

$*^\bullet$ – mulţimea succesor (mulţime de tranziţii, respectiv poziţii) a mulţimii $*$ (mulţime de poziţii, respectiv tranziţii)

$\sigma = t_{i_1} t_{i_2} ... t_{i_k}$ – secvenţă de executări de tranziţii fără precizarea vectorilor de marcaj accesaţi

$\sigma = t_{i_1} M_1 t_{i_2} M_2 ... t_{i_k} M_k$ – secvenţă de executări de tranziţii, cu precizarea vectorilor de marcaj accesaţi

$\overline{\sigma} \in \mathbb{Z}^n$ – vectorul numărului de executări de tranziţii din secvenţa σ

$\overline{\sigma}(t_i)$ – numărul de executări ale tranziţiei t_i din secvenţa σ

$R(M_0)$, $R(\boldsymbol{N}, M_0)$ – mulţimea de accesibilitate corespunzătoare unui marcaj iniţial dat

$L(M_0)$, $L(\boldsymbol{N}, M_0)$ – mulţimea tuturor secvenţelor de executări de tranziţii, pornind dintr-un marcaj iniţial dat.

$M_0[\sigma > M_k$ – vectorul de marcaj M_k este accesibil din vectorul de marcaj M_0 prin secvenţa de executări de tranziţii σ

$K(p_j)$ – capacitatea poziţiei p_j

ω – simbol utilizat în construcţia arborilor sau grafurilor de acoperire pentru reţelele nemărginite

$C(t_{i1}, t_{i2})$ – capacitatea de jetoane dintre tranziţiile t_{i1}, t_{i2}

$\boldsymbol{A}^- \in \mathbb{Z}^{n\times m}$ – matricea de incidență de intrare asociată unei rețele Petri

$\boldsymbol{A}^+ \in \mathbb{Z}^{n\times m}$ – matricea de incidență de ieşire asociată unei rețele Petri

$\boldsymbol{A} = \boldsymbol{A}^+ - \boldsymbol{A}^- \in \mathbb{Z}^{n\times m}$ – matricea de incidență asociată unei rețele Petri

$\boldsymbol{u}_k$ – vector de executare (de control)

C – circuit orientat

$M(C_k)$ – numărul total de jetoane aflate pe circuitului C_k în marcajul M

$M_0(C_k)$ – numărul total de jetoane aflate pe circuitului C_k în marcajul inițial M_0

$D(C_k)$ – întârzierea totală pe circuitul C_k

Ω – spațiul eşantioanelor (mulțimea tuturor realizărilor posibile ale unui experiment)

$\mathbb{E}$ – spațiul evenimentelor

P – funcția de probabilitate

$(\Omega, \mathbb{E}, P)$ – câmp de probabilitate

$P[A \mid B]$ – probabilitatea condițională de apariție a evenimentului A în ipoteza că s-a produs evenimentul B

X – variabilă aleatoare

F – funcția de repartiție a unei variabile aleatoare

p – densitatea de repartiție a unei variabile aleatoare discrete

f – densitatea de repartiție a unei variabile aleatoare continue

$\mathrm{M}[X]$ – valoarea medie a variabilei aleatore X

$\mathrm{Var}[X]$ – *dispersia* (*varianța*) variabilei aleatoare X

σ_X –deviație standard a variabilei aleatoare X

$\oplus$ – operație definită pe $\mathbb{R} \cup \{-\infty\}$ prin $a \oplus b = \max\{a, b\}$

$\otimes$ – operație definită pe $\mathbb{R} \cup \{-\infty\}$ prin $a \otimes b = a + b$

Capitolul 1

Introducere

La nivelul introductiv pe care şi-l propune acest capitol, convenim ca, în funcție de contextul exprimării, prin *sistem cu evenimente discrete* să înțelegem fie un sistem real, fie un model matematic (ce descrie funcționarea unui sistem real), a cărui evoluție este raportată la apariția unor evenimente. Astfel, producerea evenimentelor joacă rolul de *cauză* pentru dinamica sistemului şi are drept *efect* modificare stărilor sistemului, evidențiind o certă similitudine cu aşa-numita „tratare pe stare" a sistemelor continue sau discrete în timp. Mai mult chiar, şi în cazul unui sistem cu evenimente discrete se poate vorbi despre o *funcție de tranziție a stărilor*, care formalizează riguros faptul că sistemul trece dintr-o stare în alta numai ca urmare a producerii unui eveniment şi că sistemul păstrează starea în care se află până la producerea unui nou eveniment.

Analogia cu sistemele continue sau discrete în timp, pe care le vom referi sub numele de „sisteme clasice" trebuie însă utilizată concomitent cu înțelegerea corectă şi completă a deosebirilor privind interpretarea cauzală a comportării. Dacă în cazul sistemelor clasice, cauzele şi efectele sunt valorile unor semnale, care, cel puțin sub raport teoretic, prin variații acoperă intervale (adică mulțimi cu aceeaşi cardinalitate ca $\mathbb{R}$), în cazul sistemelor cu evenimente discrete, *mulțimea evenimentelor* ce pot apărea, precum şi *mulțimea stărilor* în care poate tranzita sistemul sunt *discrete* (adică au cel mult cardinalitatea lui $\mathbb{N}$). Dacă dinamica sistemelor clasice este raportată la un *ceas sincron* ce măsoară scurgerea uniformă a timpului (continuu, sau discret - eşantionat cu o anumită perioadă), dinamica sistemelor cu evenimente discrete se raportează la un *ceas asincron*, care marchează succesiunea evenimentelor şi, nu în mod obligatoriu, momentele de producere a lor. În cosecință, modelele de tip ecuații diferențiale sau cu diferențe ce populează literatura sistemelor clasice s-au dovedit neadecvate pentru descrierea dinamicilor pilotate de evenimente, motiv pentru care a fost necesar să se recurgă la instrumente matematice de altă factură.

Sistemele cu evenimente discrete s-au individualizat ca direcţie proprie de cercetare în ultimii 10 – 15 ani, având un impact considerabil asupra dezvoltării tehnologice din diverse arii ale ingineriei, cum ar fi: sisteme de fabricaţie, sisteme de transport, sisteme de comunicaţii, sisteme de operare şi platforme software dedicate, precum şi asupra controlului de tip procedural a numeroase clase de procese automatizate. În perioada mai sus amintită, domeniul *Sistemelor cu evenimente discrete* s-a fondat dintr-un nucleu interdisciplinar, construit prin aportul a grupuri de cercetători cu formaţii ştiinţifice diferite şi alimentat de o serie de resurse distincte din spaţiul informaticii teoretice şi a matematicilor aplicate, dintre care cele mai importante ar fi: *teoria automatelor şi a limbajelor formale*, *teoria reţelelor Petri*, *teoria sistemelor de aşteptare*, *teoria algebrică a sincronizării*, *analiza perturbaţiilor* (Cao and Ho, 1990). În acest context heterogen, este dificil şi poate chiar hazardat a localiza, în timp, un anumit moment sau o anumită lucrare, de ale căror semnificaţii să se lege actul de naştere al noului domeniu. Există însă câteva contribuţii de pionierat care au precizat ferm conexiunile cu descrierile şi principiile generale ce guvernează sistemele dinamice şi care, astfel, au deschis perspectiva ca ansamblul preocupărilor din domeniu să poată fi încadrat în amplul edificiu al Ştiinţei sistemelor, drept o nouă entitate, recte *Sisteme cu evenimente discrete*. Dintre respectivele lucrări amintim: (Ho and Cassandras, 1983), (Cohen et al., 1985), (Ramadge and Wonham, 1987).

Astăzi, în urma unui proces de maturizare alert, însoţit de numeroase succese pe plan teoretic şi aplicativ, domeniul *Sistemelor cu evenimente discrete* a dobândit o recunoaştere generală ca direcţie de cercetare pe deplin conturată, figurând la poziţia 93C65 în clasificarea comună realizată în anul 2000 de către prestigioasele publicaţii „Zentralblatt MATH” şi „Mathematical Reviews”. Cu toate acestea, istoria relativ recentă a domeniului face însă ca nici în prezent să nu dispunem de un suport teoretic unificat, capabil de a asigura compatibilitatea între metodologiile de sorginte matematică diferită, enumerate în paragraful anterior. În sensul unei atare unificări, paşi notabili au fost realizaţi de (Bacelli, 1992), (Cassandras, 1993), (Cassandras et al., 1995), (Lewis et al., 1995).

Cum era de aşteptat, drumul până la conturarea unui punct de vedere teoretic atotcuprinzător scoate la lumină multiple dificultăţi în construirea punţilor de legătură, un rol însemnat jucându-l şi inerenta atitudine de conservare a tradiţiilor de cercetare manifestată de diferite colective. Deşi la nivel general, atare punţi deja există, preferinţa pentru un anumit instrument de investigare este uzual motivată de specificul problematicii şi de experienţa acumulată în abordarea acelei problematici.

Concomitent cu progresul *Sistemelor cu evenimente discrete* ca domeniu de cercetare, acest domeniu şi-a făcut apariţia şi ca obiect de studiu în planurile de învăţământ ale universităţilor occidentale, americane şi japoneze, fiind binecunoscut

dinamismul lor în preluarea şi diseminarea noutăţilor ştiinţifice. Urmare a heterogenităţii domeniului însuşi, cursurile respective se caracterizează printr-o varietate destul de mare a modalităţilor de tratare. Totodată, şedinţele de aplicaţii ce acompaniază orele de predare a noţiunilor teoretice sunt orientate pe diferite medii software ce permit simularea, analiza şi sinteza sistemelor cu evenimente discrete. Fiecare dintre aceste medii exploatează un anumit suport teoretic, actualmente existând instrumente fiabile, produse de firme cu experienţă, care se bazează pe teoria automatelor (ca de exemplu State-Flow (MathWorks, 1997)), pe teoria reţelelor Petri (ca de exemplu software-urile prezentate în (Mortensen, 2003)), pe teoria sistemelor de aşteptare (ca de exemplu ARENA (Kelton et al., 2002), pe teoria algebrică a sincronizării (ca de exemplu Scilab (Cohen et al., 2001)).

Pornind de la realitatea curentă a domeniului (atât din perspectiva ştiinţifică cât şi didactică), am considerat că elaborarea unei lucrări cu caracter monografic, care să asigure, deopotrivă, accesibilitatea lecturii pentru cititorii nefamiliarizaţi cu problematica studiată, trebuie să permită o incursiune cât mai completă în universul *Sistemelor cu evenimente discrete*, realizată gradual, ghidată într-o manieră unică sub raportul informaţiilor şi al limbajului de exprimare. Astfel, am decis să structurăm materialul pe fundamentul teoretic al reţelelor Petri, care, pe parcursul celor patru decenii scurse de la prezentarea tezei de doctorat a matematicianului german Carl Adam Petri (Petri, 1962), au demonstrat o deosebită flexibilitate în abordarea a numeroase tipuri de probleme practice, precum şi o mare capacitate de extindere ca sferă de operare, prin înglobarea unor puncte de vedere tot mai complexe.

Reţelele (grafurile) propuse în (Petri, 1962) dispuneau de un mecanism capabil de a guverna, pe principiile algebrei boolene, evoluţia unui vector cu elemente numere naturale, având semnificaţie de stare, fără precizarea momentelor efective de timp când aveau loc modificările stării. Cercetările ulterioare au condus şi la încorporarea informaţiilor temporale (Ramamoorthy and Ho, 1980), (Ajmone Marsan et al., 1985), astfel încât, în prezent, pentru studierea sistemelor cu evenimente discrete, avem la dispoziţie modele de tip reţea Petri netemporizată (care permit studii calitative) şi, respectiv, de tip reţea Petri temporizată (care permit studii cantitative). Introducerea temporizării s-a realizat de aşa manieră încât să permită nuanţările de model determinist sau stohastic, binecunoscute din cazul sistemele clasice.

Aceste rafinări ale conceptului iniţial formulat de Petri (rafinări care nu includ, în totalitate, extinderile propuse în literatură) evidenţiază atât resursele oferite pentru modelare, cât şi compatibilitatea cu alte instrumente şi tipuri de modele. Reţelele Petri pot modela fenomenele specifice sistemelor cu evenimente discrete, cum ar fi succesiunea (o evoluţie succede alteia), alegerea sau conflictul (selectarea uneia din mai multe posibilităţi de evoluţie), concurenţa (startarea unor evoluţii paralele), sincronizarea (încheierea unor evoluţii paralele), excluderea mutuală (condiţionarea reciprocă a unor evoluţii), care pot fi

formulate în contexte temporizate sau netemporizate. Pe de altă parte, informaţiile conţinute de alte modalităţi de descriere a dinamicii sistemelor cu evenimente discrete (automate, sisteme de aşteptare, reprezentări de stare max-plus) pot fi grefate cu uşurinţă pe arhitectura modelelor de tip reţea Petri. În plus, studiul reţelelor Petri este uzual acompaniat de o abordare la nivel vizual, prin reprezentări grafice expresive. Drept urmare, literatura de specialitate raportează o largă utilizare a reţelelor Petri în modelare, analiză şi proiectare, acoperind o arie semnificativă de procese controlate secvenţial, de la dinamica unor entităţi individuale (David et Alla, 1992), (Zurawski and Zhou, 1994), la dinamica unor entităţi colective (sisteme mari – eng. *large scale systems*), ca de exemplu, sisteme hardware şi software (Peterson, 1981), (Levi and Agrawala, 1990), procese chimice (Yamalidou et al., 1990), sisteme de fabricaţie (Desrochers and Al-Jaar, 1993), (Zhou and DiCesare, 1993), (Lewis et al., 1995), roboţi şi sisteme de transport (Freedman, 1991), (Cassandras et al., 1995), sisteme de comunicaţii (Nissanke, 1997).

Lucrarea noastră caută să ilustreze pe parcursul a opt capitole cele mai importante dintre aspectele punctate anterior. Modul de înlănţuire al capitolelor a avut în vedere creşterea treptată a complexităţii problematicii, precum şi posibilitatea stabilirii unor conexiuni relevante inter-capitole. S-a reuşit, astfel, o acoperire a elementelor definitorii pentru formalismul reţelelor Petri şi utilizarea acestuia în practică, contextului netemporizat fiindu-i alocate primele cinci capitole, iar contextului temporizat, următoarele trei capitole.

Contextul netemporizat tratează modelarea, analiza şi proiectarea din perspectiva dinamicilor coordonate doar la nivel logic, momentele de apariţie a evenimentelor fiind ordonate ca succesiune, dar fără precizarea lor concretă şi fără utilizarea unor durate concrete de timp. În context temporizat, modelarea, analiza şi proiectarea includ informaţii concrete privind momentele şi/sau duratele de timp, considerate de natură deterministă sau stohastică.

În linii mari vorbind, proprietăţile calitative ce fac obiectul tehnicilor de analiză şi sinteză pe modele netemporizate sunt robuste în raport cu orice variaţie de factură temporală şi se referă la funcţionarea ansamblului sistemic fără incidente de procedură tehnică (într-o manieră repetabilă, fără aglomerări sau blocări în realizarea serviciilor). Pe de altă parte, analiza şi sinteza pe modele temporizate vizează proprietăţile cantitative în sensul de indici de performanţă asociabili unor criterii de eficienţă economică (de tipul servicii realizate în unitatea de timp, grad de ocupare a prestatorilor de servicii etc.).

Fiecare din cele opt capitole conţine un breviar teoretic urmat de un număr de aplicaţii ce exemplifică conceptele, rezultatele şi metodele din breviar. În aplicaţii se face apel atât la reţele Petri fără semnificaţii de modele ale unor dinamici reale (ce permit exersarea noţiunilor teoretice), cât şi la reţele Petri care modelează funcţionarea unor sisteme fizice (ce asigură ancorarea cunoştinţelor teoretice pe terenul intuiţiei

fenomenologice). Pentru aceleaşi sistemele fizice sunt construite iniţial modele netemporizate concentrând atenţia asupra proprietăţilor comportamentale şi structurale, iar ulterior se adaugă informaţii legate de timp, spre a utiliza modelele temporizate în investigaţii cantitative. Soluţiile prezentate pentru aplicaţii au în vedere, deopotrivă, varianta analitică şi cea asistată de calculator; în general, pentru aplicaţiile de complexitate redusă sunt date rezolvări analitice, pentru aplicaţiile de complexitate medie sunt date rezolvări analitice şi asistate de calculator, iar pentru aplicaţiile de complexitate mare sunt date numai rezolvări asistate de calculator.

Pentru studiile prin simulare precum şi pentru rezolvarea asistată de calculator a aplicaţiilor se utilizează software-ul ***Petri Net Toolbox*** (Mahulea et al., 2001), (Matcovschi et al., 2002), (Matcovschi et al., 2003), proiectat, implementat şi testat în cadrul Catedrei de Automatică şi Informatică Industrială a Facultăţii de Automatică şi Calculatoare din Iaşi. Acest software a fost anume conceput pentru exploatare sub mediul MATLAB care este familiar instruirii în Automatică. Ajuns deja la versiunea 2.0, el oferă o interfaţă cu utilizatorul foarte prietenoasă ce asigură definirea reţelelor sub formă grafică, simularea cu facilităţi multiple de animaţie şi accesul la instrumente de analiză şi sinteză ce acoperă toate dezvoltările teoretice din prezenta lucrare.

Ultimul capitol al lucrării realizează o trecere în revistă a principalelor caracteristici ale software-ului ***Petri Net Toolbox***, în limba engleză, urmărind structura de informaţii disponibilă în Help-ul on-line al produsului. După cunoştinţele noastre, ***Petri Net Toolbox*** este singurul produs disponibil sub MATLAB care utilizează reţelele Petri în explorarea sistemelor cu evenimente discrete, exceptând două pachete elaborate recent, şi anume (Svádová and Hanzálek, 2001), care nu posedă interfaţă grafică proprie, utilizând interfaţa software-ului Petri-Maker (Mortensen, 2003), şi (Iordache and Antsaklis, 2002), care nu are nici un fel de interfaţă grafică, topologiile reţelelor fiind descrise direct matriceal.

Elaborată în spiritul tendinţelor actuale de cercetare şi educaţie din domeniul sistemelor cu evenimente discrete, cartea are un pronunţat caracter de originalitate pentru literatura tehnică din România, raportată la lucrările existente cu tematică asemănătoare (Stănescu et al., 1996), (Leţia şi Aştilean, 1998), (Jucan şi Ţiplea, 1999), (Mânzu şi Cernega, 2001). Acest caracter rezultă din ponderea acordată aplicaţiilor, din maniera lor de prezentare detaliată (menită să ilustreze atât tratarea analitică, cât şi modul de implicare a software-ului) şi din rolul jucat de ***Petri Net Toolbox*** ca instrument ce extinde substanţial aria de exploatare a MATLAB-ului. Totodată precizăm faptul că tehnicile ilustrate prin aceste aplicaţii nu fac uz de întreaga instrumentaţie pusă la dispoziţie de teoria reţelelor Petri, ci, ca urmare a unei selecţii judicioase, se concentrează pe acele abordări a căror eficienţă a fost validată de un număr mare de implementări practice, raportate în literatură. Consecvenţi ideii de a menţine accesibilitatea şi complexitatea noţiunilor teoretice

la un nivel mediu, am procedat la excluderea unor tematici mai dificile, care sunt referite frecvent drept extensii ale formalismului de bază al reţelelor Petri, cum ar fi: reţele Petri colorate (pentru care recomandăm cititorului lucrările (Jensen, 1992, 1994, 1997)), reţele Petri interpretate, continue şi hibride (pentru care recomandăm cititorului lucrarea (David et Alla, 1992)), reţele Petri în supervizare (pentru care recomandăm cititorului lucrarea (Antsaklis and Moody, 1998).

În conceperea, structurarea şi realizarea acestei cărţi, o contribuţie majoră a avut lucrarea (Pastravanu, 1997) pe baza căreia s-au organizat cursul şi şedinţele de aplicaţii la disciplina *Sisteme dinamice cu evenimente discrete* de la Facultatea de Automatică şi Calculatoare din Iaşi. Şedinţele de aplicaţii respective au oferit şi cadrul necesar pentru verificarea robusteţii software-ului ***Petri Net Toolbox*** şi pentru validarea pe o baterie largă de teste a algoritmilor implementaţi.

Capitolul 2

Modele de tip rețea Petri netemporizată

Breviar teoretic

BT2.1. Conceptul de rețea Petri netemporizată

O *rețea Petri* (eng. *Petri net*) se compune dintr-un tip particular de *graf orientat* notat ***N*** şi o *stare inițială* M_0, denumită *marcaj inițial* (eng. *initial marking*).

Graful ***N*** al rețelei Petri este orientat, ponderat şi bipartit, constând din două tipuri de noduri, denumite *poziții* sau *locații* (eng. *place*) şi respectiv *tranziții* (eng. *transition*); *arcele orientate* (eng. *arc*) unesc fie o poziție cu o tranziție, fie o tranziție cu o poziție. Nu există arce care să conecteze două poziții între ele, sau două tranziții între ele. Ca simbolizare grafică, pozițiile se reprezintă prin cercuri, iar tranzițiile prin bare sau dreptunghiuri. Arcele sunt etichetate cu ponderile lor (eng. *weight*) (valori întregi, pozitive); un arc cu ponderea *k* poate fi privit ca o mulțime de *k* arce paralele cu pondere unitară. Etichetele pentru pondere unitară se omit în reprezentările grafice uzuale.

Un *marcaj* sau o *stare* atribuie fiecărei poziții un număr întreg mai mare sau egal cu 0. Dacă un marcaj atribuie poziției *p* întregul $k \geq 0$, se spune că *p* este marcată cu *k jetoane* (eng. *token*). Din punct de vedere grafic, în cercul corespunzător poziției *p* se vor plasa *k* discuri. Orice marcaj *M* este *un vector coloană m–dimensional*, unde *m* notează numărul total al pozițiilor. Componenta *i* a vectorului $M = [M(p_1)\ M(p_2) \ldots M(p_m)]^T$, notată $M(p_i)$ reprezintă numărul de jetoane din poziția p_i. Din motive de concizie a scrierii, în unele capitole ale acestei cărți un marcaj *M* va fi, de asemenea, reprezentat prin *m*-uplul $M = (M(p_1),\ M(p_2), \ldots, M(p_m))$.

Aspectele prezentate anterior se formalizează matematic prin următoarea definiție.

O *rețea Petri* este un cvintuplu, $\boldsymbol{PN} = (P, T, F, W, M_0)$ în care:

- $P = \{p_1, p_2, \ldots, p_m\}$ este mulțimea pozițiilor sau locațiilor (finită);
- $T = \{t_1, t_2, \ldots, t_n\}$ este mulțimea tranzițiilor (finită);
- $F \subseteq (P \times T) \bigcup (T \times P)$ este mulțimea arcelor;
- $W : F \to \{1, 2, 3, \ldots\}$ este funcția de ponderare a arcelor;
- $M_0 : P \to \{0, 1, 2, 3, \ldots\}$ este funcția de marcaj inițial.

Câteva comentarii sunt necesare pentru a aprofunda detaliile acestei formalizări:

1. Mulţimile *P* şi *T* sunt disjuncte, $P \bigcap T = \varnothing$.
2. Pentru a asigura obiectul definiţiei de mai sus, mulţimile *P* şi *T* satisfac condiţia $P \bigcup T \neq \varnothing$.
3. Definiţia funcţiei de ponderare se poate extinde pe mulţimea tuturor perechilor ordonate din $(P \times T) \bigcup (T \times P)$, considerând $W : (P \times T) \bigcup (T \times P) \to \{0,1,2,3,...\}$, cu observaţia că, pentru acele perechi care nu sunt în mulţimea *F*, valoarea funcţiei *W* este 0 şi aceste perechi nu sunt reprezentate grafic. Mulţimea *F* corespunde perechilor a căror pondere este nenulă şi numai acestea sunt reprezentate grafic.
4. Definiţia unei reţele Petri consideră implicit că toate poziţiile reţelei pot conţine un număr oricât de mare de jetoane. Se spune că reţeaua este cu *capacitate infinită*.
5. O structură de reţea Petri $\boldsymbol{N} = (P, T, F, W)$ fără nici o specificaţie referitoare la marcaj se va nota cu $\boldsymbol{N}$, notaţie care desemnează *topologia reţelei*.
6. O reţea Petri cu un marcaj iniţial M_0 se va nota prin $(\boldsymbol{N}, M_0)$.
7. O reţea Petri cu un marcaj oarecare *M* se va nota prin $(\boldsymbol{N}, M)$.

În problemele de modelare ce utilizează conceptele de *condiţii* şi *evenimente*, poziţiile reprezintă condiţii şi tranziţiile reprezintă evenimente. O tranziţie (eveniment) posedă un număr de poziţii de intrare şi ieşire, care reprezintă *pre-condiţii* şi respectiv *post-condiţii* pentru evenimentul în cauză. Prezenţa unui jeton într-o poziţie trebuie înţeleasă ca valoare logică „adevărat” pentru condiţia asociată respectivei poziţii.

BT2.2. Terminologie uzuală

Dacă o poziţie *p* este atât poziţie de intrare, cât şi de ieşire pentru o tranziţie *t*, atunci *p* şi *t* formează o *buclă autonomă* (eng. *self-loop*). O reţea Petri care nu conţine bucle autonome se numeşte *pură* (eng. *pure*). O buclă autonomă poate fi întotdeauna transformată într-o buclă neautonomă prin adăugarea simultan a unei poziţii şi a unei tranziţii formale (eng. *dummy*). Orice reţea impură poate fi transformată într-o reţea pură.

O reţea Petri se numeşte *ordinară* (eng. *ordinary*) dacă toate arcele sale au pondere unitară. Dacă într-o reţea Petri există cel puţin un arc a cărui pondere este mai mare decât 1, atunci se spune că reţeaua respectivă este *generalizată*.

Fie *F* mulţimea tuturor arcelor unei reţele Petri $\boldsymbol{N}$. Se numesc *mulţime predecesor* (eng. *pre-set*) şi *mulţime succesor* (eng. *post-set*) *a tranziţiei t* (fig. BT2.2.1.(a)) două mulţimi de poziţii definite prin:

$${}^{\bullet}t = \{p \mid (p,t) \in F\} = \text{mulţimea tuturor poziţiilor de intrare ale lui } t;$$

şi, respectiv, prin:

$$t^{\bullet} = \{p \mid (t,p) \in F\} = \text{mulţimea tuturor poziţiilor de ieşire ale lui } t.$$

Se numesc *mulţime predecesor* şi *mulţime succesor a poziţiei p* (fig. BT2.2.1.(b)) două mulţimi de tranziţii definite prin:

$${}^{\bullet}p = \{t \mid (t,p) \in F\} = \text{mulţimea tuturor tranziţiilor de intrare ale lui } p;$$

şi, respectiv, prin:

$$p^{\bullet} = \{t \mid (p,t) \in F\} = \text{mulţimea tuturor tranziţiilor de ieşire ale lui } p.$$

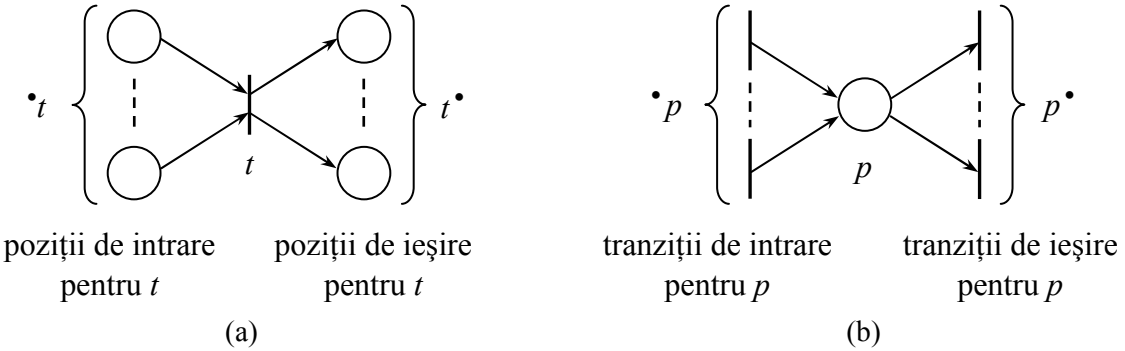


Fig. BT2.2.1. *Ilustrarea grafică a mulțimilor predecesor şi succesor*
(a) *Mulțimile de poziții* $^\bullet t$ *şi* $t^\bullet$; (b) *Mulțimile de tranziții* $^\bullet p$ *şi* $p^\bullet$.

Acelaşi mod de notare, şi anume $^\bullet\{*\}$, $\{*\}^\bullet$, se utilizează pentru a desemna mulțimea predecesor şi, respectiv, succesor a unei *mulțimi de tranziții* sau a unei *mulțimi de poziții*, notată generic $\{*\}$. Evident, mulțimile predecesor şi succesor ale unei mulțimi de tranziții sunt două mulțimi de poziții, iar mulțimile predecesor şi succesor ale unei mulțimi de poziții sunt două mulțimi de tranziții.

BT2.3. Validarea şi executarea tranzițiilor în rețele cu capacitate infinită – evoluția stărilor

Marcajul unei rețele Petri are semnificația de *stare a rețelei* şi se poate modifica în conformitate cu următorul procedeu denumit *regula tranziției* (validare şi executare) (ilustrare în **AP2.1**).

a) Se spune că o tranziție t este *validată* (eng. *enabled*) dacă fiecare poziție de intrare (predecesor) p a lui t este marcată cu cel puțin $W(p,t)$ jetoane, unde $W(p,t)$ notează ponderea arcului de la p la t.
b) O tranziție validată poate sau nu să fie *executată* sau *declanşată* (eng. *fired*), după cum evenimentul asociat tranziției are sau nu loc.
c) Executarea unei tranziții validate îndepărtează $W(p,t)$ jetoane din fiecare poziție de intrare (predecesor) p a lui t şi adaugă $W(t,p)$ jetoane la fiecare poziție de ieşire (succesor) p a lui t, unde $W(t,p)$ este ponderea arcului de la t la p.

O tranziție fără nici o poziție de intrare se numeşte *tranziție sursă* (eng. *source*). O tranziție fără nici o poziție de ieşire se numeşte *tranziție receptor* (eng. *sink*). Modul de operare al acestor tranziții este următorul:

a) O tranziție sursă este necondiționat validată (fără a fi obligatoriu ca să se execute). Executarea ei produce jetoane.
b) Executarea unei tranziții receptor consumă jetoane, fără a produce jetoane.

În teoria rețelelor Petri netemporizate se consideră că *executarea unei tranziții nu consumă timp* şi că *jetoanele pot rămâne în poziții pentru orice durată de timp* (oricât de mică sau oricât de mare). Întrucât executarea unei tranziții este instantanee, se consideră că tranzițiile se execută *numai secvențial*, adică nu se poate vorbi de două tranziții executate

simultan (sau în paralel). Aceste presupuneri fac ca modelul de tip rețea Petri netemporizată să fie utilizat numai pentru investigarea proprietăților logice, calitative, care nu depind de timp.

Pentru o rețea Petri $\boldsymbol{N}$ cu un marcaj inițial M_0, urmărind executarea secvențială a tranzițiilor, se pot determina marcajele succesive ale rețelei. Procesul de modificare a marcajului (stării) rețelei poate fi descris într-o manieră sintetică printr-un arbore sau printr-un graf (diferit de graful rețelei Petri!), ce poartă denumirea de *arbore*, respectiv, *graf de accesibilitate* (eng. *reachability tree/graph*) (ilustrare în **AP2.4**). Aceste concepte sunt detaliate în BT3.3 din capitolul 3, cu referire la situația mai generală a *arborelui/grafului de acoperire* (eng. *coverability tree/graph*).

BT2.4. Unele extensii pentru rețele Petri netemporizate

BT2.4.1. Rețele Petri cu capacitate finită

Pentru regula de validare a unei tranziții prezentată anterior s-a presupus că fiecare poziție are *capacitate infinită*. În modelarea sistemelor fizice este firesc a considera o limită superioară a numărului de jetoane pe care îl poate conține fiecare poziție, asociind fiecărei poziții p *capacitatea* sa (eng. *capacity*), notată $K(p)$, definită ca numărul maxim de jetoane ce pot fi conținute în p. O astfel de rețea se numeşte cu *capacitate finită*.

Într-o rețea cu capacitate finită, pentru *validarea* unei tranziții t este necesară următoarea *condiție suplimentară*: numărul de jetoane în fiecare poziție de ieşire p a lui t nu poate să depăşească capacitatea poziției respective, $K(p)$, atunci când t s-ar executa (ilustrare în **AP2.1**). În acest caz, regula tranziției se va numi *regula strictă a tranziției* spre a o deosebi de cea enunțată în paragraful BT2.3, care mai este uneori referită drept *regula simplă a tranziției*.

Fiind dată o rețea Petri de capacitate finită ($\boldsymbol{N}$, M_0) este posibil de aplicat fie regula strictă a tranziției direct pentru rețeaua ($\boldsymbol{N}$, M_0), fie regula simplă a tranziției pentru o *rețea transformată adecvat*, notată $(\boldsymbol{N}', M_0')$. Presupunând că rețeaua $\boldsymbol{N}$ este pură, următorul *algoritm* permite construcția rețelei $(\boldsymbol{N}', M_0')$ plecând de la ($\boldsymbol{N}$, M_0), prin *metoda pozițiilor complementare*:

Pas 1. Pentru *fiecare* poziție p, se adaugă o poziție complementară p', al cărei marcaj inițial este dat de $M_0'(p) = K(p) - M_0(p)$.

Pas 2. Între *fiecare* tranziție t şi *unele poziții complementare* p' se trasează arce suplimentare, (t,p') sau (p',t), cu ponderile $W(t, p') = W(p,t)$, respectiv $W(p',t) = W(t,p)$, astfel încât suma jetoanelor în poziția p şi în poziția complementară corespunzătoare p' să fie egală cu capacitatea $K(p)$, atât înainte, cât şi după executarea tranziției t (adică să se asigure satisfacerea condiției $M(p) + M'(p') = K(p)$).

Subliniem faptul că *metoda nu adaugă* tranziții suplimentare (ilustrare în **AP2.2**). Grafurile de accesibilitate ale rețelelor ($\boldsymbol{N}$, M_0) şi $(\boldsymbol{N}', M_0')$ sunt *izomorfe* în sensul că prezintă aceleaşi secvențe posibile de executare a tranzițiilor.

BT2.4.2. Rețele cu probabilități şi priorități

În unele modele de tip rețea Petri două sau mai multe tranziții modelează evenimente dintre care unul şi numai unul se poate produce la un moment dat (în *conflict*). Implicit se consideră

că probabilitățile de apariție a acestor evenimente sunt egale. În cazul în care această presupunere nu este în conformitate cu sistemul fizic modelat, probabilitățile de apariție a evenimentelor în conflict pot fi asignate în mod explicit ca probabilități de executare a tranzițiilor care modelează evenimentele respective (ilustrare în **AP2.5**).

De asemenea, atunci când se dorește a impune modul de alegere a tranziției care urmează a fi executată dintre două sau mai multe tranziții validate simultan se poate utiliza o *rețea Petri cu priorități* care constă dintr-o rețea Petri obișnuită și o relație de ordine parțială între tranzițiile rețelei (ilustrare în **AP2.3**).

BT2.4.3. Rețele cu arce inhibitoare

Introducerea arcelor *inhibitoare* extinde capacitatea de modelare a rețelelor Petri. Un arc inhibitor conectează o poziție la o tranziție și are rolul de a inversa logica de validare și executarea a acesteia. Tranziția respectivă este validată numai dacă numărul de jetoane din poziția de intrare a arcului inhibitor este strict mai mic decât ponderea arcului. Arcul inhibitor se reprezintă grafic printr-un segment ce conectează cele două noduri, având un mic cerc la capătul dinspre tranziția inhibată. Nu există arce inhibitoare care conectează o tranziție la o poziție (ilustrare în **AP2.3**).

BT2.5. Modelarea cu rețele Petri ordinare

BT2.5.1. Structuri tipice utilizate în modelare

Structurile tipice utilizate în modelarea sistemelor cu evenimente discrete prin intermediul rețelelor Petri sunt prezentate sintetic în fig. BT2.5.1.

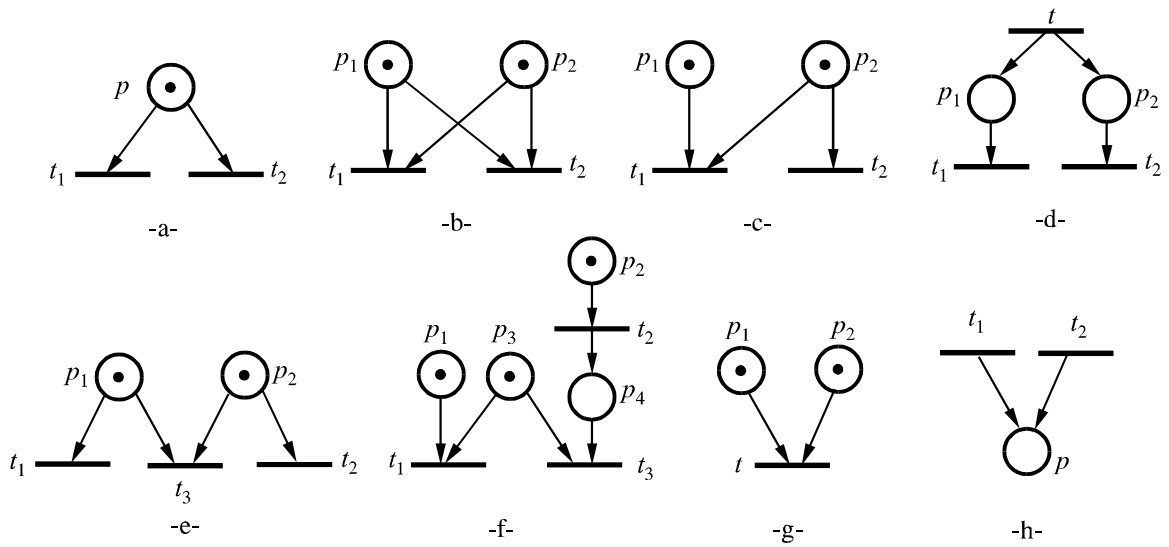


Fig. BT2.5.1. *Structuri tipice utilizate în modelarea cu rețele Petri*

(a) *Conflict, decizie sau alegere liberă;* (b) *Alegere liberă extinsă;*
(c) *Alegere asimetrică;* (d) *Paralelism sau concurență;* (e) *Confuzie simetrică;*
(f) *Confuzie asimetrică;* (g) *Sincronizare;* (h) *Post-condiție comună.*

BT2.5.2. Capacitatea de modelare a unor subclase de rețele Petri ordinare

Rețelele Petri ordinare pot fi organizate în *subclase* definite pe baza unor considerente *topologice*. Apartenența unei rețele la o anumită subclasă este reflectată în *capacitatea de modelare*, în sensul că rețeaua nu va putea conține toate *structurile tipice* definite în secțiunea precedentă, ci numai o parte din acestea (în funcție de subclasa căreia îi aparține). Această partajare pe subclase va servi, în secțiunea BT3.5 din capitolul 3, la formularea unor *tehnici dedicate* analizei anumitor proprietăți comportamentale pentru aceste tipuri de rețele.

Maşina de stare (eng. *state machine*) este o rețea Petri ordinară în care fiecare tranziție t are o singură poziție de intrare şi o singură poziție de ieşire. Formalizând, avem:

$$\forall t \in T: \quad |{}^\bullet t| = |t^\bullet| = 1, \tag{BT2.5.1}$$

unde $|*|$ notează cardinalitatea (numărul de elemente al) mulțimii $*$.

Referindu-ne la structurile tipice ilustrate în fig. BT2.5.1, se constată că o maşină de stare *conține numai* alegeri libere şi post-condiții comune (ilustrare în **AP2.5**).

Graful marcat (eng. *marked graph*) sau graful de evenimente (eng. *event graph*) este o rețea Petri ordinară în care fiecare poziție p are o singură tranziție de intrare şi o singură tranziție de ieşire. Formalizând, avem

$$\forall p \in P: \quad |{}^\bullet p| = |p^\bullet| = 1. \tag{BT2.5.2}$$

În termenii structurilor tipice ilustrate în fig. BT2.5.1, se constată că un graf marcat *conține numai* concurențe şi sincronizări (ilustrare în **AP2.4** şi **AP2.5**).

Termenul de graf marcat sau graf de evenimente se datorează faptului că acest tip de rețea Petri poate fi reprezentată printr-un *graf orientat unipartit* (care posedă un singur tip de noduri!) în care *nodurile* corespund tranzițiilor (adică evenimentelor), *arcele* corespund pozițiilor, iar jetoanele se plasează *pe arce* (adică arcele sunt marcate) (ilustrare în **AP2.4**). Astfel regula tranziției se aplică nodurilor grafului orientat, o executare constând, în această reprezentare grafică, în îndepărtarea unui jeton de pe fiecare din arcele de intrare ale nodului în cauză şi adăugarea unui jeton pe fiecare din arcele de ieşire.

Rețeaua cu alegeri libere (eng. *free choice net*) este o rețea Petri ordinară în care orice arc ce iese dintr-o poziție p este caracterizat prin una din următoarele două situații:

(i) este *singurul* arc care pleacă din p,

sau:

(ii) este *singurul* arc care intră într-o anumită tranziție (adică în acea tranziție nu mai intră nici un alt arc, în afara celui ce pleacă din p).

Avem următoarea formalizare:

$$\forall p \in P: \quad |p^\bullet| \le 1 \quad \text{sau} \quad {}^\bullet(p^\bullet) = \{p\}. \tag{BT2.5.3}$$

Această descriere matematică este echivalentă cu implicația:

$$\forall p_1, p_2 \in P: \quad p_1^\bullet \cap p_2^\bullet \neq \varnothing \quad \Rightarrow \quad |p_1^\bullet| = |p_2^\bullet| = 1. \tag{BT2.5.4}$$

Condiția (BT2.5.4) trebuie privită ca o relaxare a condiției (BT2.5.2). Totodată, condiția (BT2.5.4) relaxează şi condiția (BT2.5.1).

Examinând aceste particularități prin prisma structurilor tipice ilustrate în fig. BT2.5.1, se constată că o rețea cu alegeri libere *conține numai* alegeri libere, post-condiții comune, concurențe şi sincronizări.

Rețeaua cu alegeri libere extinse (eng. *extended free-choice net*) este o rețea Petri ordinară în care este satisfăcută condiția:

$$\forall p_1, p_2 \in P: \quad p_1^\bullet \cap p_2^\bullet \neq \varnothing \Rightarrow p_1^\bullet = p_2^\bullet . \qquad \text{(BT2.5.5)}$$

Condiția (BT2.5.5) trebuie privită ca o relaxare a condiției (BT2.5.4).

În termenii structurilor tipice discutate, ilustrate în fig. BT2.5.1, se constată că o rețea cu alegeri libere extinse *conține numai* alegeri libere, alegeri libere extinse, post-condiții comune, concurențe şi sincronizări.

Rețeaua cu alegeri asimetrice (eng. *asymmetric–choice net*), sau rețeaua *simplă*, este o rețea Petri în care este satisfăcută condiția:

$$\forall p_1, p_2 \in P: \quad p_1^\bullet \cap p_2^\bullet \neq \varnothing \Rightarrow (p_1^\bullet \subseteq p_2^\bullet \text{ sau } p_1^\bullet \supseteq p_2^\bullet) . \qquad \text{(BT2.5.6)}$$

Condiția (BT2.5.6) trebuie privită ca o relaxare a condiției (BT2.5.5).

Referindu-ne la structurile tipice ilustrate în fig. BT2.5.1, se constată că o rețea cu alegeri asimetrice *conține numai* alegeri libere, alegeri libere extinse, alegeri asimetrice, post-condiții comune, concurențe, sincronizări şi confuzii asimetrice (ilustrare în **AP3.5** şi **AP3.6**).

Examinând definițiile subclaselor de rețele Petri ordinare formulate mai sus, se constată că între acestea există relațiile de incluziune reprezentate grafic în fig. BT2.5.2.

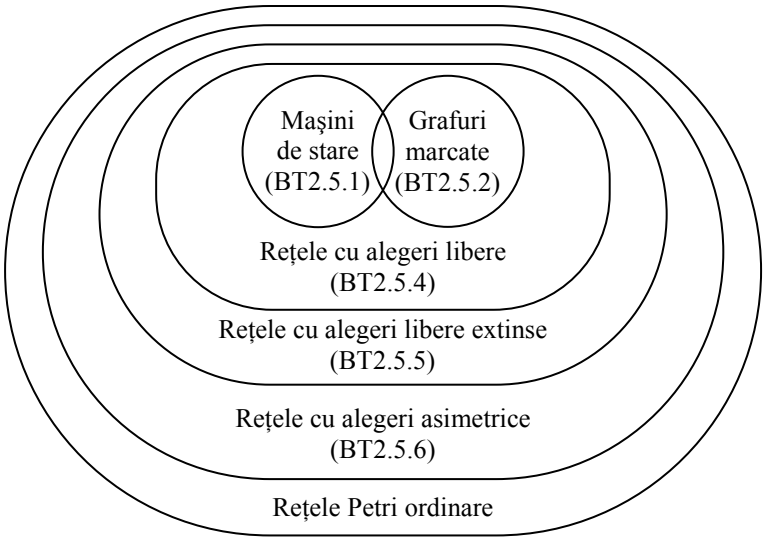


Fig. BT2.5.2. *Relațiile de incluziune între subclasele de rețele Petri ordinare.*

Tabelul BT2.5.1. sintetizează informaţiile referitoare la capacitatea de modelare a diferitelor subclase de reţele Petri ordinare în raport cu structurile tipice ce au fost prezentate în secţiunea BT2.5.1.

Tab. BT2.5.1. *Prezentare sintetică a capacităţii de modelare a diferitelor subclase de reţele Petri.*

Structură / **Clasă**	**Alegeri libere**	**Alegeri libere extinse**	**Alegeri asimetrice**	**Concu-renţe**	**Confuzii simetrice**	**Confuzii asimetrice**	**Sincro-nizări**	**Post-condiţii comune**
Maşină de stare	DA	–	–	–	–	–	–	DA
Graf marcat	–	–	–	DA	–	–	DA	–
Reţea cu alegeri libere	DA	–	–	DA	–	–	DA	DA
Reţea cu alegeri libere extinse	DA	DA	–	DA	–	–	DA	DA
Reţea cu alegeri asimetrice	DA	DA	DA	DA	–	DA	DA	DA
Reţea Petri ordinară	DA	DA	DA	DA	DA	DA	DA	DA

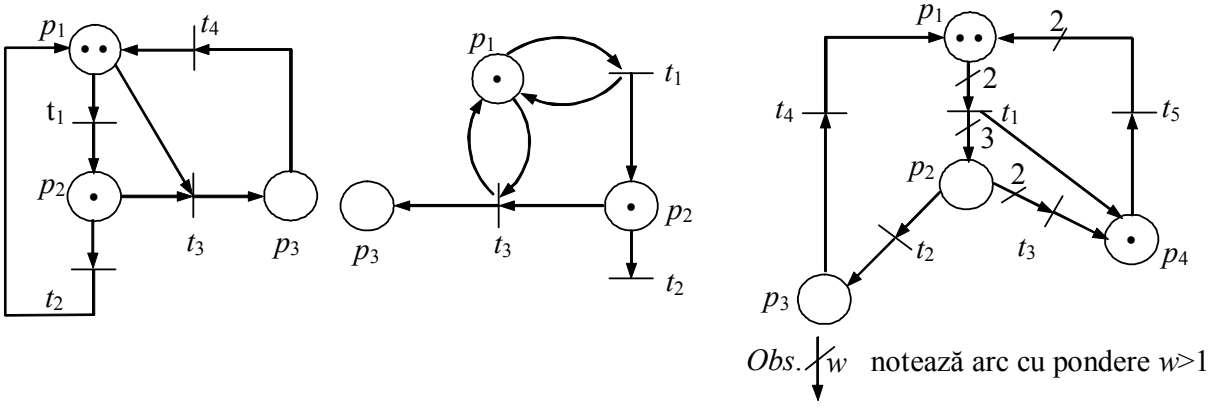
Capacitatea de modelare creşte în sensul acceptării de noi structuri tipice, această creştere fiind perfect compatibilă cu relaţiile de incluziune stabilite între subclase în paragraful anterior (vezi şi fig. BT2.5.2). Dar, după cum va reieşi din secţiunea următoare, creşterea capacităţii de modelare face ca criteriile de analiză a proprietăţilor comportamentale să devină tot mai complexe şi mai dificil de aplicat. Din acest motiv, insistăm asupra necesităţii elaborării unor modele cât mai puţin sofisticate, care să fie capabile să surprindă acele detalii de funcţionare ce trebuie avute în vedere pe parcursul analizei sau proiectării.

Aplicaţii

AP2.1.

Pentru reţelele Petri cu topologia şi marcajul iniţial indicat în fig. AP2.1.1, să se precizeze care dintre tranziţii sunt validate şi marcajul ce rezultă după executarea fiecăreia din aceste tranziţii în cazurile următoare:

1. reţelele au capacităţi infinite;
2. reţelele au capacităţi finite după cum urmează:
   (i) pentru reţeaua din fig. AP2.1.1.(a): $K(p_1)=K(p_3)=2$, $K(p_2)=1$;
   (ii) pentru reţeaua din fig. AP2.1.1.(b): $K(p_1)=K(p_2)=K(p_3)=1$;
   (iii) pentru reţeaua din fig. AP2.1.1.(c): $K(p_1)=2$, $K(p_2)=3$, $K(p_3)=K(p_4)=1$.



(a) (b) (c)

Fig. AP2.1.1. *Rețelele Petri studiate în AP2.1.*

Soluție

1. Aplicând regula simplă a tranziției rețelei din fig. AP2.1.1.(a) cu marcajul inițial $M_0=(2,1,0)$ se obține:

- tranziția t_1 este validată deoarece $M_0(p_1)>W(p_1,t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1=(1,2,0)$;
- tranziția t_2 este validată deoarece $M_0(p_2)=W(p_2,t_2)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2=(3,0,0)$;
- tranziția t_3 este validată deoarece $M_0(p_1)>W(p_1,t_3)$ și $M_0(p_2)=W(p_2,t_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3=(1,0,1)$.

Analog, pentru rețeaua din fig. AP2.1.1.(b) cu marcajul inițial $M_0=(1,1,0)$ rezultă:

- tranziția t_1 este validată deoarece $M_0(p_1)>W(p_1,t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1=(1,2,0)$;
- tranziția t_2 este validată deoarece $M_0(p_2)=W(p_2,t_2)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2=(3,0,0)$;
- tranziția t_3 este validată deoarece $M_0(p_1)>W(p_1,t_3)$ și $M_0(p_2)=W(p_2,t_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3=(1,0,1)$.

Pentru rețeaua din fig. AP2.1.1.(c) cu marcajul inițial $M_0=(2,0,0,1)$, aplicarea regulii simple a tranziției conduce la:

- tranziția t_1 este validată deoarece $M_0(p_1)>W(p_1,t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1=(0,3,0,2)$;
- tranziția t_5 este validată deoarece $M_0(p_4)=W(p_4,t_5)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2=(4,0,0,0)$.

2. Ținând cont de marcajele la care s-ar ajunge prin executarea tranzițiilor validate determinate la punctul 1 şi de capacitățile finite precizate ale pozițiilor, prin aplicarea regulii stricte a tranziției se obține:

(i) pentru rețeaua din fig. AP2.1.1.(a), singura tranziție validată este t_3 deoarece $M_0(p_1) > W(p_1,t_3)$, $M_0(p_2) = W(p_2,t_3)$ şi $K(p_3) > M_0(p_3) + W(t_3,p_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3 = (1,0,1)$;

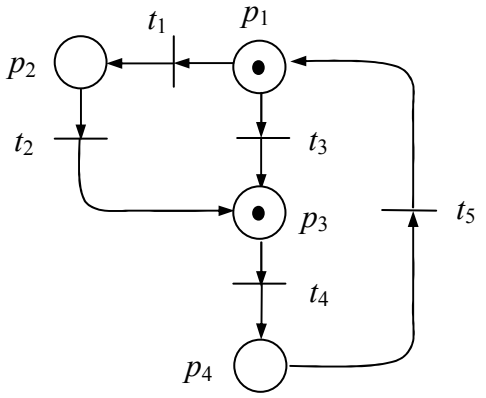
(ii) pentru rețeaua din fig. AP2.1.1.(b) sunt validate:

- tranziția t_2 deoarece $M_0(p_2) = W(p_2,t_2)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2 = (3,0,0)$;
- tranziția t_3 deoarece $M_0(p_1) > W(p_1,t_3)$, $M_0(p_2) = W(p_2,t_3)$ şi $K(p_3) = M_0(p_3) + W(t_3,p_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3 = (1,0,1)$;

(iii) pentru rețeaua din fig. AP2.1.1.(c) nici una dintre tranziții nu este validată.

AP2.2.

Transformați rețeaua Petri cu capacitate finită din fig. AP2.2.1, într-o rețea Petri cu capacitate infinită.



$$K(p_3) = 1$$
$$K(p_1) = K(p_2) = K(p_4) = \infty$$

Fig. AP2.2.1. *Rețeaua Petri cu capacitate finită studiată în AP2.2.*

Soluție

Utilizând metoda pozițiilor complementare cu referire la poziția p_3, se adaugă poziția p_3' astfel încât să fie satisfăcute următoarele relații:

$$W(t_3,p_3) = W(p_3',t_3) = 1,$$
$$W(t_3,p_3') = W(p_3,t_3) = 1,$$
$$M_0(p_3') = K(p_3) - M_0(p_3) = 1 - 1 = 0.$$

Rețeaua Petri cu capacitate finită din fig. AP2.2.1 este echivalentă cu rețeaua Petri de capacitate infinită prezentată în fig. AP2.2.2.

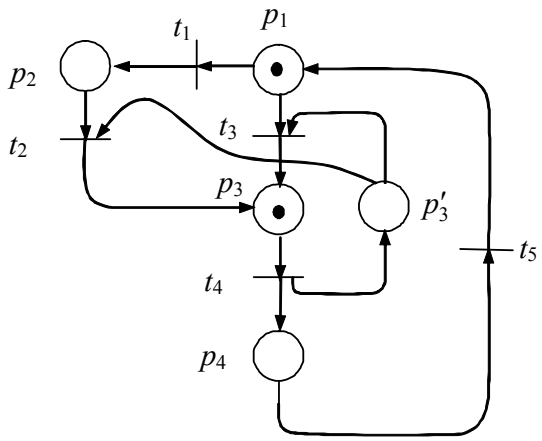


Fig. AP2.2.2. *Rețeaua Petri cu capacitate infinită corespunzătoare celei din fig. AP2.2.1.*

AP2.3.

Se consideră rețeaua Petri din fig. AP2.3.1 care modelează un sistem cu un producător (subrețeaua din partea stângă) şi un consumator (subrețeaua din partea dreaptă).

1. Să se explice modul de operare al rețelei.
2. Se consideră şi un al doilea consumator, care este autorizat să consume numai atunci când primul consumator nu solicită produs. Să se construiască modelul tip rețea Petri al sistemului cu doi consumatori.

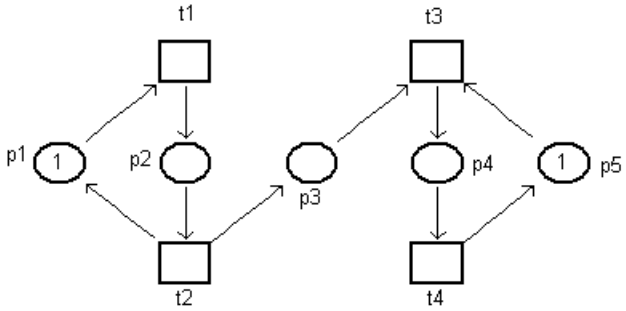


Fig. AP2.3.1. *Modelul de tip rețea Petri al unui sistem cu un producător şi un consumator.*

Soluție

1. În poziția p_3 se acumulează un număr de jetoane semnificând produsele furnizate de către producător. Pe măsură ce consumatorul are nevoie de aceste produse, se va executa tranziția t_3, câte o dată pentru fiecare produs preluat de către consumator.

2. Prezența celui de-al doilea consumator se modelează printr-o subrețea identică cu cea utilizată pentru primul consumator. În cazul conectării celor două subrețele de tip consumator

ca în fig. AP2.3.2.(a), în lipsa altor precizări, cei doi consumatori vor avea drepturi egale de a consuma, ceea ce înseamnă că modelul nu răspunde în totalitate cerinţelor din enunţ. Pentru a include în model şi informaţiile referitoare la disciplina de consum, se va asigna prioritate 0 tranziţiei t_3 şi, respectiv, prioritate 1 tranziţiei t_5.

O variantă echivalentă cu alocarea priorităţilor o constituie controlul tranziţiei t_5 printr-un arc inhibitor conform figurii AP2.3.2.(b); astfel, dacă p_3 conţine jeton/jetoane, t_5 va fi validată numai când p_5 nu conţine jeton (adică primul consumator nu solicită produs).

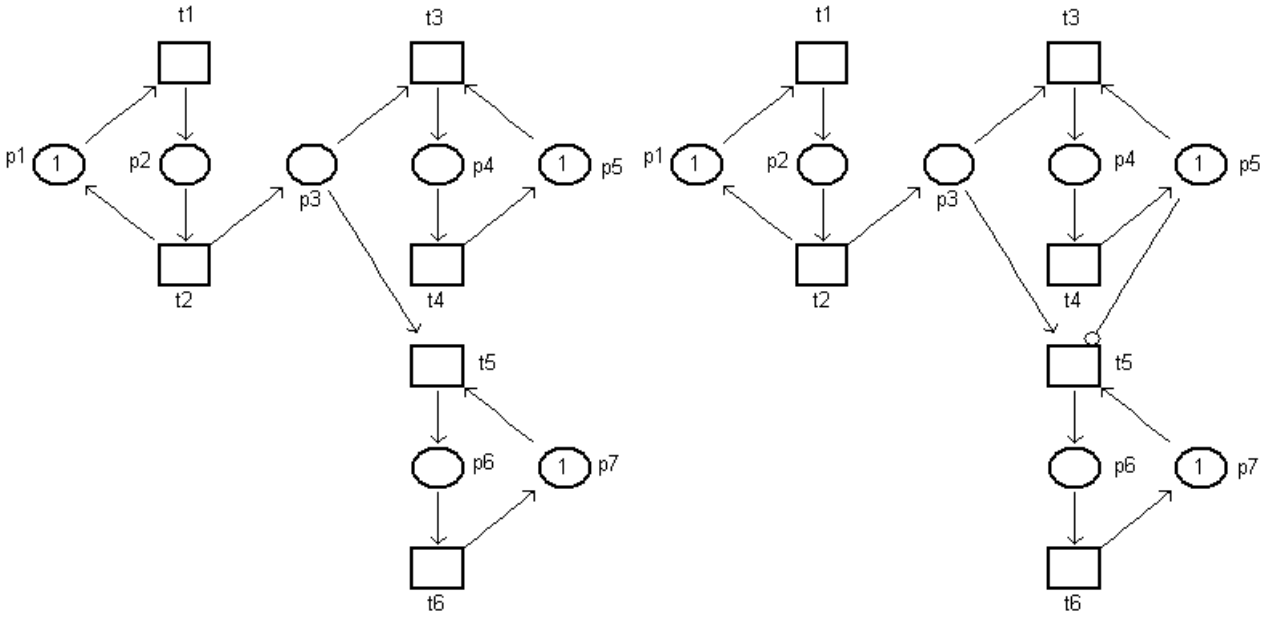


Fig. AP2.3.2. *Modelul de tip reţea Petri al sistemului cu un producător şi doi consumatori:*
(a) *asignând tranziţiei t_3 prioritate mai mare decât tranziţiei t_5;*
(b) *utilizând arc inhibitor pe tranziţia t_5.*

AP2.4.

Se consideră un protocol de comunicaţii, adaptat după (Desrochers and Al-Jaar, 1993), reprezentat prin modelul din fig. AP2.4.1. Semnificaţiile poziţiilor şi tranziţiilor sunt prezentate în tabelele AP2.4.1, respectiv AP2.4.2.

1. Să se precizeze succesiunea de executări de tranziţii care, plecând din marcajul iniţial din fig. AP2.4.1, asigură revenirea la acest marcaj.
2. Să se specifice succesiunea de evenimente corespunzătoare transmiterii complete a unui mesaj.
3. Să se precizeze clasa de reţele Petri căreia îi aparţine acest model.

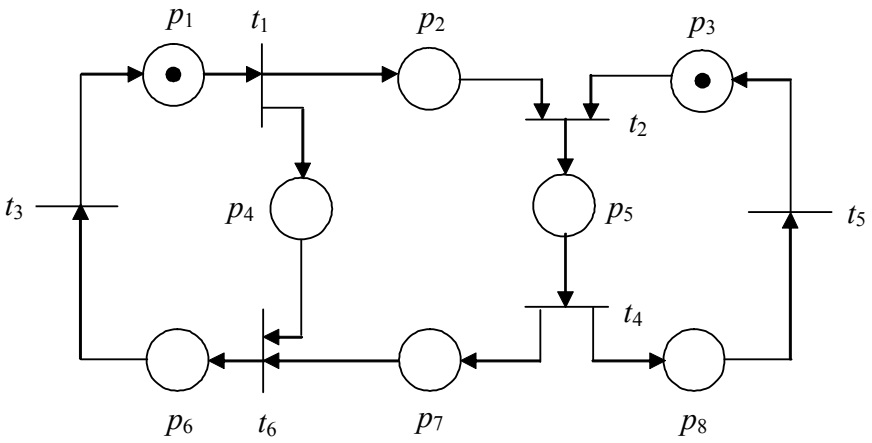


Fig. AP2.4.1. *Modelul de tip rețea Petri al unui protocol de comunicații.*

Tab. AP2.4.1. *Semnificațiile pozițiilor din modelul prezentat în fig. AP2.4.1.*

Poziție	**Condiție**
p_1	Emițătorul (E) pregăteşte mesajul.
p_2	Buffer-ul de comunicație (B) conține mesajul.
p_3	Receptorul (R) este pregătit pentru recepție.
p_4	(E) aşteaptă semnalul de confirmare (*acknowledge* – ACK).
p_5	(R) primeşte mesajul.
p_6	(E) primeşte semnalul de confirmare (ACK).
p_7	(B) conține semnalul de confirmare (ACK).
p_8	(R) procesează mesajul primit.

Tab. AP2.4.2. *Semnificațiile tranzițiilor din modelul prezentat în fig. AP2.4.1.*

Tranziție	**Eveniment**
t_1	(E) începe transmisia mesajului.
t_2	(R) începe primirea mesajului
t_3	(E) termină de primit semnalul de confirmare (ACK) şi începe pregătirea unui nou mesaj.
t_4	(R) termină primirea mesajului şi începe transmisia semnalului de confirmare (ACK) şi procesarea mesajului.
t_5	(R) termină procesarea şi începe pregătirea pentru recepționarea unui nou mesaj.
t_6	(E) începe primirea semnalului de confirmare (ACK).

Soluție

1. Arborele de accesibilitate furnizat de ***Petri Net Toolbox*** în mod grafic şi semnificația marcajelor sunt prezentate în fig. AP2.4.2. Se poate observa că, plecând din marcajul inițial M_0, succesiunea de executări de tranziții $\sigma = t_1 t_2 t_4 t_5 t_6 t_3$ asigură revenirea la acesta.

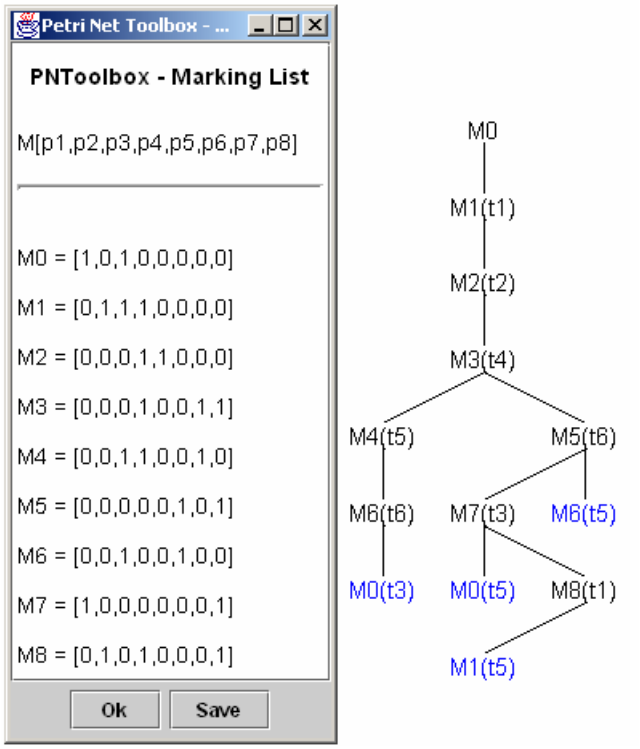


Fig. AP2.4.2. *Arborele de accesibilitate pentru rețeaua Petri din fig. AP2.4.1.*

2. Pe baza semnificației fizice a tranzițiilor din modelul studiat, succesiunii de tranziții σ determinată la punctul precedent îi corespunde următoarea succesiune de evenimente care au loc la transmiterea completă a unui mesaj:

- (E) începe transmisia mesajului (t_1).
- (E) începe primirea semnalului de confirmare (ACK) (t_2).
- (R) începe primirea mesajului (t_4).
- (R) termină primirea mesajului şi începe transmisia semnalului ACK şi procesarea semnalului (t_5).
- (R) termină procesarea şi începe pregătirea pentru recepționarea unui nou mesaj (t_6).
- (E) termină de primit semnalul ACK şi începe pregătirea unui nou mesaj (t_3).

Observație: Se poate observa că, în afară de secvența σ precizată anterior, mai există încă două secvențe de executări de tranziții care, plecând din marcajul inițial, asigură revenirea la acesta, şi anume $\sigma' = t_1 t_2 t_4 t_6 t_5 t_3$ şi $\sigma'' = t_1 t_2 t_4 t_6 t_3 t_5$. Deoarece duratele operațiilor care au loc în sistemul fizic nu sunt implicate în modelul logic reprezentat de rețeaua Petri netemporizată, nu se poate preciza ordinea în care se vor executa tranzițiile t_5 şi t_6. (Pentru interpretarea în context temporizat, vezi **AP6.2**.)

3. Rețeaua Petri prezentată în fig. AP2.4.1 face parte din clasa grafurilor marcate Datorită tipului particular de rețea, drept reprezentare grafică se poate utiliza un graf unipartit, conform fig. AP2.4.3.

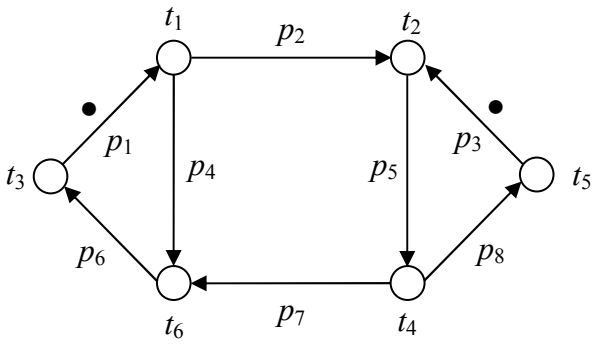


Fig. AP2.4.3. *Modelul de tip graf marcat al protocolului de comunicații reprezentat sub formă de graf orientat unipartit.*

AP2.5.

Se consideră rețeaua Petri având topologia din fig. AP2.5.1. Pozițiile au capacitățile $K(p_1)=1$ şi $K(p_2)=1$.

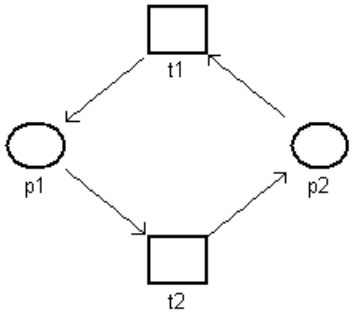


Fig. AP2.5.1. *Topologia rețelei Petri utilizată în AP2.5.*

1. Pentru marcajul inițial $M_0(p_1)=1$ şi $M_0(p_2)=0$, să se execute simularea în varianta ***Step*** pentru 4 evenimente înregistrând rezultatele simulării într-un fişier de tip jurnal. Se va citi jurnalul şi se va preciza *secvența de executări de tranziții* care are loc pe parcusul simulării precum şi *marcajele* prin care trece rețeaua.
2. Să se reia punctul 1, schimbând marcajul inițial în $M_0'(p_1)=0$ şi $M_0'(p_2)=1$, fără a modifica topologia rețelei.
3. Să se precizeze clasa de rețele Petri căreia îi aparține acest model.
4. Să se justifice faptul că rețeaua Petri din fig. AP2.5.1 modelează un server cu două stări I (*iddle*) şi W (*working*).

5. Să se precizeze cum trebuie modificată reţeaua astfel încât să poată modela un server cu posibilităţi de defectare, situaţie în care va mai exista o stare D (*down*). Se presupune că serverul nu se poate defecta decât în timpul lucrului. Se vor avea în vedere următoarele două situaţii:
   (i) Servirea clientului aflat în lucru în momentul defectării poate fi reluată după remediere.
   (ii) Servirea clientului aflat în lucru în momentul defectării nu mai poate fi reluată, trecându-se la servirea unui nou client.
6. Să se simuleze în varianta ***Run Fast*** modelele de la punctul 5, pentru un număr total de 10.000 de evenimente, în următoarele situaţii:
   (i) Probabilitatea defectării serverului este egală cu a funcţionării corecte;
   (ii) Probabilitatea defectării serverului (10%) este mult mai mică decât cea a funcţionării corecte (90%);
   (iii) Probabilitatea defectării serverului (90%) este mult mai mare decât cea a funcţionării corecte (10%).

   Să se precizeze prin ce diferă modelele de simulare de la subpunctele (i), (ii) şi (iii) ale punctului 6. Pentru fiecare caz în parte să se precizeze următorii indicatori de performanţă ai serverului: numărul total de clienţi sosiţi la server care au început să fie serviţi, numărul total de clienţi serviţi complet şi numărul total de defectări ale serverului. Corelaţi aceste rezultate cu parametrii modelului folosit pentru simulare.

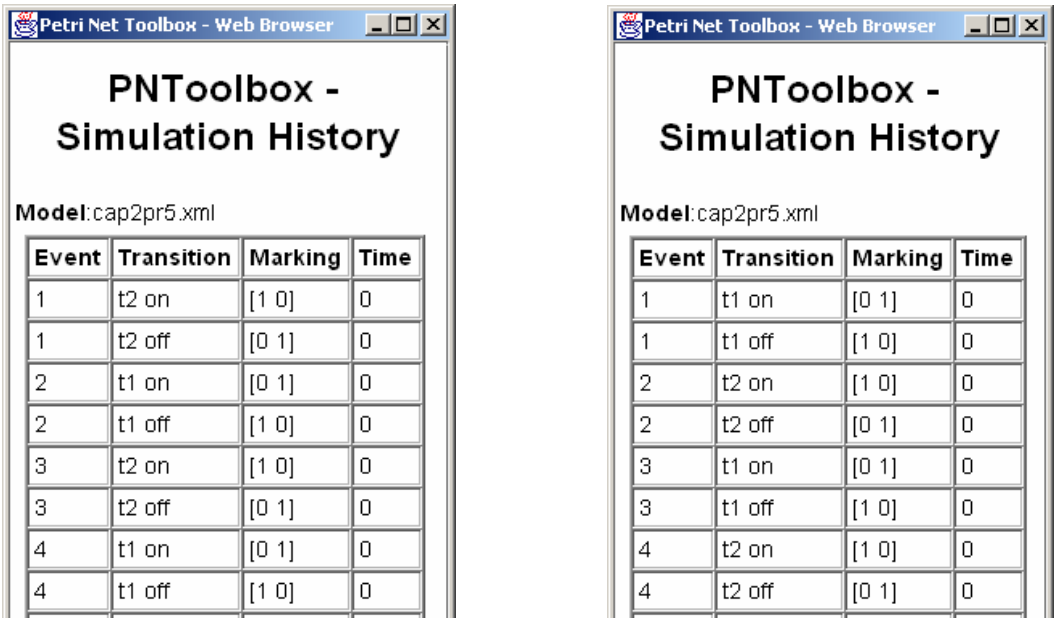
Soluţie

1. În urma efectuării unui experiment de simulare în modurile ***Step*** şi ***Run Slow*** a funcţionării unei reţele Petri, mediul ***Petri Net Toolbox*** oferă posibilitatea de a înregistra, într-un fişier de tip jurnal, secvenţa de tranziţii care a fost executată pe parcursul simulării şi succesiunea de marcaje prin care a trecut reţeaua.

Utilizând această facilitate pentru simularea apariţiei unui număr de 4 evenimente în reţeaua Petri din fig. AP2.5.1 cu marcajul iniţial $M_0 = (1 \quad 0)$, se înregistrează rezultatele prezentate în fig. AP2.5.2.(a). Se observă că secvenţa de executări de tranziţii care are loc este $\sigma = t_2, t_1, t_2, t_1$. Există numai două marcaje distincte prin care trece reţeaua (succesiv), anume $M_0 = (1 \quad 0)$ şi $M_1 = (0 \quad 1)$.

2. Păstrând topologia reţelei dar schimbând marcajul iniţial în $M_0' = (0 \quad 1)$, în urma simulării cu ajutorul mediului ***Petri Net Toolbox*** a apariţiei unui număr de 4 evenimente se obţin rezultatele prezentate în fig. AP2.5.2(b). În acest caz, secvenţa de executări de tranziţii care are loc este: $\sigma' = t_1, t_2, t_1, t_2$. Se observă că şi în acest caz există numai două marcaje distincte prin care trece reţeaua, anume $M_0' = (0 \quad 1)$ şi $M_1' = (1 \quad 0)$.

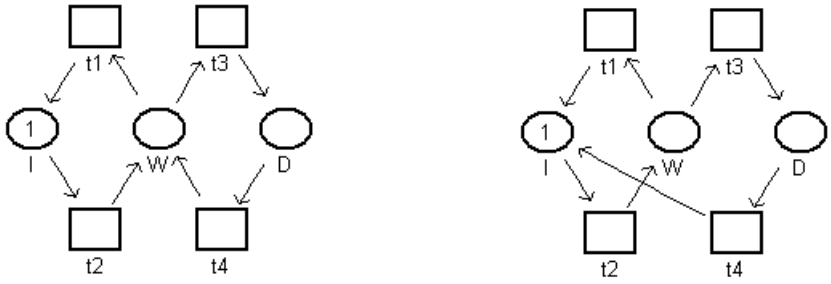
3. Reţeaua Petri ordinară cu topologia din fig. AP2.5.1 este atât *graf marcat*, pentru că fiecare poziţie are o singură tranziţie de intrare şi o singură tranziţie de ieşire, cât şi *maşină de stare*, pentru că fiecare tranziţie are o singură poziţie de intrare şi o singură poziţie de ieşire.


Petri Net Toolbox - Web Browser
PNToolbox - Simulation History
Model:cap2pr5.xml
Event | Transition | Marking | Time
1 | t2 on | [1 0] | 0
1 | t2 off | [0 1] | 0
2 | t1 on | [0 1] | 0
2 | t1 off | [1 0] | 0
3 | t2 on | [1 0] | 0
3 | t2 off | [0 1] | 0
4 | t1 on | [0 1] | 0
4 | t1 off | [1 0] | 0
Petri Net Toolbox - Web Browser
PNToolbox - Simulation History
Model:cap2pr5.xml
Event | Transition | Marking | Time
1 | t1 on | [0 1] | 0
1 | t1 off | [1 0] | 0
2 | t2 on | [1 0] | 0
2 | t2 off | [0 1] | 0
3 | t1 on | [0 1] | 0
3 | t1 off | [1 0] | 0
4 | t2 on | [1 0] | 0
4 | t2 off | [0 1] | 0

(a) (b)

Fig. AP2.5.2. *Jurnalul furnizat de Petri Net Toolbox după simularea apariției unui număr de 4 evenimente în rețeaua Petri din fig. AP2.5.1:* (a) *cu marcajul inițial* $M_0 = (1 \quad 0)$; (b) *cu marcajul inițial* $M_0' = (0 \quad 1)$.

4. După cum am văzut la punctele 1 şi 2, la apariția unui eveniment (executarea unei tranziții) rețeaua Petri își schimbă marcajul (starea). În ambele situații considerate există numai două marcaje distincte, (1, 0) şi (0, 1), în care rețeaua evoluează succesiv, ca urmare a executării uneia dintre cele două tranziții. Distincția dintre cele două marcaje este dată de prezența jetonului din marcajul inițial în una dintre pozițiile rețelei. Din aceste motive, rețeaua Petri din fig. AP2.5.1 modelează un sistem fizic cu două stări (corespunzătoare celor două marcaje). Un asemenea sistem ar putea reprezenta, de exemplu, un server cu două stări şi anume starea în care aşteaptă sosirea unui client (*iddle* – I) – corespunzătoare marcajului (1, 0) – şi cea în care serveşte un client (*working* – W) – corespunzătoare marcajului (0, 1).



(a) (b)

Fig. AP2.5.3. *Topologiile rețelelor Petri în cazul unui server cu trei stări.*

5. Pentru a modifica rețeaua Petri din fig. AP2.5.1 astfel încât să poată modela un server cu posibilități de defectare, este necesară introducerea unei poziții suplimentare care va fi asociată stării în care serverul este defect (*down* – D). Rețelele Petri corespunzătoare celor două situații (i)

şi (ii) sunt prezentate în fig. AP2.5.3, ambele având topologii de maşină de stare. Semnificațiile fizice ale pozițiilor şi tranzițiilor sunt prezentate în tab. AP2.5.1, respectiv AP2.5.2.

Tab. AP2.5.1. *Semnificațiile pozițiilor din modelul prezentat în fig. AP2.4.1.*

Poziție	**Stare**
I	(S) este disponibil (aşteaptă un client).
W	(S) serveşte un client.
D	(S) este defect.

Tab. AP2.5.2. *Semnificațiile tranzițiilor din modelul prezentat în fig. AP2.4.1.*

Tranziție	**Eveniment**
t_1	Serverul (S) a terminat de servit un client.
t_2	(S) începe servirea unui nou client.
t_3	Apare un defect la (S).
t_4	Se termină remedierea serverului.

6. În mediul ***Petri Net Toolbox***, pentru ambele rețele prezentate în fig. AP2.5.3, probabilităților de executare a tranzițiilor t_3 şi t_4 li se pot asigna valorile corespunzătoare probabilităților de apariție a evenimentelor pe care le modelează (apariția unui defect şi, respectiv, terminarea remedierii serverului). În urma efectuării unui experiment de simulare, indicatorul ***Service Sum*** asociat tranzițiilor t_2, t_1 şi t_3 corespunde numărului de clienți sosiți la server, numărului de clienți serviți complet şi, respectiv, numărului de defectări ale serverului. Tab. AP2.5.3 prezintă sintetic valorile obținute în urma simulării apariției a 10.000 de evenimente pentru cele două rețele prezentate în fig. AP2.5.3 în fiecare dintre cele trei situații.

Tab. AP2.5.3. *Indicatori obținuți în urma simulării apariției a 10.000 de evenimente în situațiile considerate la punctul 6 al aplicației AP2.5*

Situație modelată / Indicator statistic	Cu reluarea servirii clientului (AP2.5.3.a)			Fără reluarea servirii clientului (AP2.5.3.b)		
	(i)	(ii)	(iii)	(i)	(ii)	(iii)
Numărul de clienți sosiți la server	2507	4527	484	3985	4765	3441
Numărul de clienți serviți complet	2507	4527	483	1955	4295	321
Numărul de defectări ale serverului	2493	473	4517	2030	470	3119

Din tabelul de mai sus se poate observa că, pentru acelaşi număr de evenimente, raportul dintre numărul de clienți serviți complet (numărul de executări ale tranziției t_1) şi numărul de defectări ale serverului (numărul de executări ale tranziției t_3) este aproximativ egal cu raportul dintre probabilitatea de funcționare corectă şi cea de defectare a serverului (asignate tranziției t_1 şi, respectiv, tranziției t_3).

AP2.6.

Un sistem de fabricație este alcătuit din două maşini M_1, M_2 şi un braț de robot R. Orice piesă brută este prelucrată mai întâi pe M_1 şi apoi pe M_2. Robotul R mută o piesă care a fost prelucrată pe M_1, de pe maşina M_1 pe maşina M_2. Nu există depozite care să preceadă M_1 sau M_2. Încărcarea unei piese pe maşina M_1 şi descărcarea unei piese de pe maşina M_2 se desfăşoară automat. Pentru transportul unei piese, aceasta se fixează pe o paletă înainte de încărcarea pe M_1 şi se desprinde de pe paletă după descărcarea de pe M_2. Paleta eliberată la descărcarea unei piese de pe M_2 este recirculată automat pentru a se fixa pe ea o nouă piesă. În fig. AP2.6.1 se prezintă schematizat procesul de prelucrare al pieselor în sistemul de fabricație.

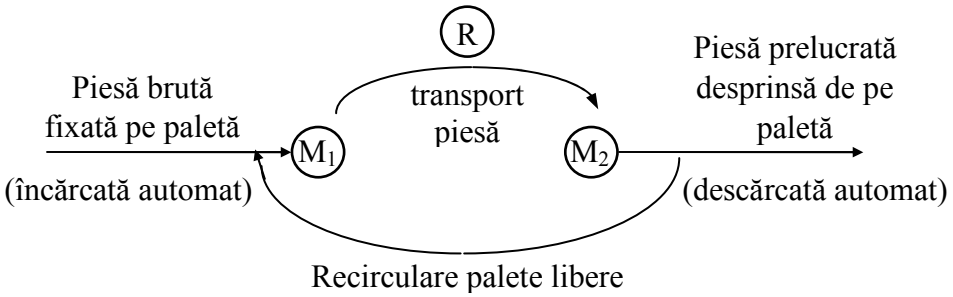


Fig. AP2.6.1. *Reprezentarea schematizată a sistemului de fabricație din AP2.6.*

Se presupune că există un număr suficient de mare de piese brute şi că M_1, M_2 şi R funcționează fără defectare.

1. Să se elaboreze un model tip rețea Petri pentru a studia comportarea de tip logic a sistemului de fabricație, atunci când se folosesc x palete pentru transport.
2. Să se simuleze, în varianta ***Step***, funcționarea sistemului de fabricație cu o singură paletă ($x = 1$) pentru un număr de 5 evenimente. Să se descrie succesiunea de stări prin care trece sistemul.
3. Se reia punctul 2 pentru $x = 4$.
4. Există un optim pentru x? Justificați.
5. Să se precizeze clasa de rețele Petri căreia îi aparține acest model.
6. Să se justifice faptul că orice piesă brută intrată în sistem ajunge să fie complet prelucrată.

Soluție

1. Modelul de tip rețea Petri al sistemului de fabricație este prezentat în fig. AP2.6.2. Numărul de jetoane din poziția p_1 în marcajul inițial este egal cu x, numărul de palete din sistemul de fabricație.

2. Succesiunea de stări prin care trece sistemul se poate obține utilizând opțiunea ***Log File***. Pentru un număr de 5 evenimente se obțin rezultatele prezentate în fig. AP2.6.3.(a).

3. Analog ca la punctul anterior utilizăm opțiunea ***Log File*** a mediului ***Petri Net Toolbox*** şi obținem succesiunea stărilor prin care trece sistemul, prezentată în fig. AP2.6.3.(b).

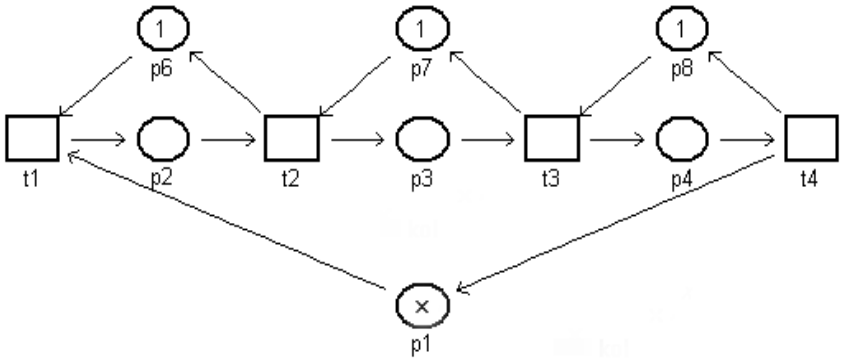


Fig. AP2.6.2. *Modelul de tip reţea Petri al sistemului de fabricaţie din fig. AP2.6.1.*

4. În sistemul de fabricaţie nu se pot găsi în acelaşi moment mai mult de 3 piese, ceea ce înseamnă că numărul optim de palete este egal cu 3. Dacă s-ar folosi $x > 3$ palete, atunci la orice moment de timp $x - 3$ palete ar fi neutilizate.

5. Reţeaua Petri din fig. AP2.6.2 este graf marcat deoarece ponderile tuturor arcelor sunt egale cu 1 şi fiecare poziţie are o singură tranziţie de intrare şi o singură tranziţie de ieşire.

6. Sistemul de fabricaţie din fig. AP2.6.1. nu conţine resurse partajate. Pentru fiecare operaţie există întotdeauna o secvenţă de executări de tranziţii care asigură eliberarea resursei necesare pentru efectuarea următoarei operaţii. În consecinţă nu se pot produce blocaje circulare de resurse, orice piesă brută intrată în sistem ajungând să fie complet prelucrată.

Petri Net Toolbox - Web Browser

PNToolbox - Simulation History

Model:cap2pr6.xml

Event	Transition	Marking	Time
1	t1 on	[1 0 0 0 1 1 1]	0
1	t1 off	[0 1 0 0 0 1 1]	0
2	t2 on	[0 1 0 0 0 1 1]	0
2	t2 off	[0 0 1 0 1 0 1]	0
3	t3 on	[0 0 1 0 1 0 1]	0
3	t3 off	[0 0 0 1 1 1 0]	0
4	t4 on	[0 0 0 1 1 1 0]	0
4	t4 off	[1 0 0 0 1 1 1]	0
5	t1 on	[1 0 0 0 1 1 1]	0
5	t1 off	[0 1 0 0 0 1 1]	0

(a)

Petri Net Toolbox - Web Browser

PNToolbox - Simulation History

Model:cap2pr6.xml

Event	Transition	Marking	Time
1	t1 on	[4 0 0 0 1 1 1]	0
1	t1 off	[3 1 0 0 0 1 1]	0
2	t2 on	[3 1 0 0 0 1 1]	0
2	t2 off	[3 0 1 0 1 0 1]	0
3	t1 on	[3 0 1 0 1 0 1]	0
3	t1 off	[2 1 1 0 0 0 1]	0
4	t3 on	[2 1 1 0 0 0 1]	0
4	t3 off	[2 1 0 1 0 1 0]	0
5	t4 on	[2 1 0 1 0 1 0]	0
5	t4 off	[3 1 0 0 0 1 1]	0

(b)

Fig. AP2.6.3. *Succesiunea de stări prin care trece reţeaua Petri din fig. AP.2.6.2. în cazul utilizării* (*a*) *unei singure palete şi* (*b*) *a unui număr de 4 palete.*

Capitolul 3

Studierea proprietăților comportamentale

Breviar teoretic

BT3.1. Definirea proprietăților comportamentale

Proprietățile comportamentale (eng. *behavioral*) ale rețelelor Petri netemporizate sunt dependente atât de topologia cât şi de marcajul inițial al rețelei. În ceea ce priveşte terminologia, este absolut necesar a face distincție față de proprietățile structurale (discutate în capitolul 5), care iau în considerare numai topologia rețelei, fiind independente de marcajul inițial al acesteia.

BT3.1.1. Accesibilitate

O secvență de executări de tranziții ale unei rețele Petri conduce la modificarea marcajului (a distribuției de jetoane), în conformitate cu aplicarea regulii tranziției. Despre un marcaj M_n se spune că este *accesibil* (eng. *reachable*) din marcajul inițial M_0 dacă există o secvență de executări de tranziții care transformă M_0 în M_n. Această secvență de executări (de tranziții) *se notează* prin: $\sigma = M_0 t_{i_1} M_1 t_{i_2} M_2 ... t_{i_k} M_k$, sau, simplu, prin: $\sigma = t_{i_1} t_{i_2} ... t_{i_k}$, când nu interesează succesiunea de marcaje. Faptul că marcajul M_n este accesibil din M_0 prin secvența de executări σ se notează $M_0[\sigma > M_n$.

Mulțimea tuturor marcajelor care pot fi atinse în rețeaua $\boldsymbol{N}$ pornind din marcajul inițial M_0 se notează prin $R(\boldsymbol{N}, M_0)$, sau, simplu, prin $R(M_0)$, atunci când se subînțelege rețeaua $\boldsymbol{N}$ la care ne referim. Mulțimea tuturor secvențelor de executare posibile în rețeaua $\boldsymbol{N}$ pornind din marcajul inițial M_0 se notează prin $L(\boldsymbol{N}, M_0)$ sau, simplu, prin $L(M_0)$ atunci când se subînțelege rețeaua $\boldsymbol{N}$ la care ne referim.

BT3.1.2. Mărginire

Se spune că o rețea Petri cu capacitate nelimitată este *k-mărginită,* sau, în limbaj prescurtat, *mărginită* (eng. *bounded*), dacă numărul de jetoane din fiecare poziție nu depăşeşte un număr finit k pentru **orice** marcaj accesibil din marcajul inițial M_0 (adică pentru orice secvență de executări de tranziții pornind de la M_0). În limbaj matematic, aceasta revine la $M(p) \le k$ pentru orice $p \in P$ şi orice $M \in R(M_0)$.

Se spune că o reţea Petri ($\boldsymbol{N}$, M_0) este *sigură* (eng. *safe*) dacă ea este 1-mărginită (ilustrare în **AP3.4**).

Din punct de vedere practic, când reţeaua modelează un proces, proprietatea de mărginire permite a studia eventualele depăşiri ale unor capacităţi fizice de procesare / memorare a informaţiei sau de prelucrare / stocare a produselor. Mărginirea asigură nedepăşirea anumitor valori, indiferent de secvenţa de evenimente (adică tranziţii executate).

BT3.1.3. Viabilitate

O reţea Petri ($\boldsymbol{N}$, M_0) se numeşte *viabilă* (eng. *live*) dacă, indiferent de marcajul care a fost atins pornind din M_0, este posibil ca, în continuare, să fie executată **orice** tranziţie t a reţelei. Până la executarea lui t poate fi necesară, eventual, executarea unui număr *finit* de alte tranziţii (ilustrare în **AP3.1**, **AP3.3**, **AP3.4**, **AP3.5** şi **AP3.6**).

Un marcaj pentru care nici o tranziţie a reţelei nu mai poate fi executată se numeşte *deadlock* (ilustrare în **AP3.2**, **AP3.5** şi **AP3.6**). În baza definiţiei viabilităţii se constată că o reţea viabilă operează fără deadlock. Pe de altă parte, o reţea care nu este viabilă, nu evoluează în mod obligatoriu către deadlock, în sensul că, pe lângă tranziţia sau tranziţiile care nu mai pot fi executate, una sau mai multe tranziţii sunt executabile de o infinitate de ori (ilustrare în **AP3.3** şi **AP4.3**). Într-un atare caz se poate vorbi despre un deadlock parţial. Nuanţări în această direcţie pot fi găsite în (Murata, 1989) şi (Păstrăvanu, 1996).

Din punct de vedere practic, când reţeaua modelează un proces, proprietatea de viabilitate permite a studia funcţionarea fără incidente nereparabile (de factură logică, adică nu defecte), care să necesite o intervenţie externă procesului.

BT3.1.4. Reversibilitate

O reţea Petri ($\boldsymbol{N}$ M_0) se spune că este *reversibilă* (eng. *reversible*), dacă pentru orice marcaj $M \in R(M_0)$, marcajul iniţial M_0 este, la rândul său, accesibil când se porneşte din M. Astfel, într-o reţea reversibilă este întotdeauna posibilă întoarcerea la marcajul iniţial (ilustrare în **AP3.1**, **AP3.4**, **AP3.5** şi **AP3.6**).

Din punct de vedere practic, când reţeaua modelează un proces, proprietatea de reversibilitate permite a studia repetabilitatea desfăşurării anumitor activităţi sau a apariţiei anumitor condiţii.

Observaţie: Mărginirea, viabilitatea şi reversibilitatea sunt proprietăţi independente una de cealaltă.

BT3.2. Producerea fenomenului de deadlock în sistemele cu resurse partajate

Exploatarea *partajată* a resurselor constituie o soluţie frecvent folosită în sistemele tehnice din diferite domenii, dintre care semnalăm drept reprezentative procesele de fabricaţie din celulele flexibile şi sistemele de operare multitasking. Prin utilizare partajată, o resursă (în general costisitoare) este întrebuinţată pentru efectuarea mai multor operaţii, dând posibilitatea ca resursa respectivă să fie alocată tuturor acestor operaţii (evident, nu simultan).

Un fenomen tipic ce poate apărea în funcţionarea sistemelor cu resurse partajate este *deadlock*-ul sau *blocarea circulară* a resurselor (ilustrare în **AP3.5** şi **AP3.6**).

Deadlock-ul se produce prin alocarea necorespunzătoare a uneia sau mai multor resurse partajate, fapt ce conduce la un lanţ circular de resurse ocupate (nu toate partajate, dar cel puţin una partajată), fiecare dintre aceste resurse aşteptând eliberarea resursei pe care o precede în lanţ. Procesul de aşteptare fiind circular, nici o resursă nu se poate elibera şi ieşirea dintr-o atare situaţie se poate realiza numai printr-o intervenţie externă sistemului. Atragem atenţia că *fenomenul de deadlock nu poate să apară în sistemele fără resurse partajate*.

Modelele tip reţea Petri netemporizată constituie un instrument foarte eficient pentru a studia cauzele şi mecanismul de producere a fenomenului de deadlock într-un context *pur logic*, fără *semnificaţii temporale*. Cu alte cuvinte, într-un sistem tehnic a cărui funcţionare presupune anumite durate de timp pentru operaţii, deadlock-ul poate să apară pentru unele durate şi poate să nu apară pentru altele. Astfel, modelul logic ne avertizează de posibila apariţie a fenomenului de deadlock, chiar dacă acesta nu se manifestă în funcţionarea fizică a sistemului datorită duratelor de timp (adică, modelul netemporizat ne avertizează că modificarea accidentală a duratelor de timp poate conduce la blocarea circulară a resurselor) (ilustrare în **AP6.7**).

BT3.3.Arbori şi grafuri de acoperire/accesibilitate

BT3.3.1. Arbori de acoperire/accesibilitate

Fiind dată o reţea Petri ($\boldsymbol{N}$, M_0), pornind de la marcajul iniţial M_0, modificarea marcajelor ca urmare a executării tranziţiilor poate fi reprezentată sub forma unui arbore, denumit *arbore de acoperire* (eng. *coverability tree*). În acest arbore, M_0 este rădăcina, iar marcajele generate sunt noduri; fiecare arc corespunde executării unei tranziţii care transformă marcajul asociat nodului de plecare în marcajul asociat nodului de sosire.

În cazul unei reţele Petri *mărginite*, arborele de acoperire se numeşte *arbore de accesibilitate* (eng. *reachability tree*), deoarece poate cuprinde *toate* marcajele accesibile pornind din marcajul iniţial M_0 (care sunt în număr finit) (ilustrare în **AP2.4**, **AP3.1**, **AP3.2**, **AP3.5** şi **AP3.6**). În această situaţie, arborele de accesibilitate poate fi utilizat pentru studierea *tuturor* proprietăţilor comportamentale. Un posibil dezavantaj îl constituie numărul mare de noduri ce poate rezulta în arborele de accesibilitate, ca urmare a complexităţii reţelei Petri studiate.

În cazul reţelelor Petri *nemărginite*, arborele de acoperire va creşte la nesfârşit. Pentru a păstra finititudinea reprezentării de tip arbore de acoperire, se introduce un simbol special ω care poate lua valori *oricât de mari*, în sensul satisfacerii următoarelor proprietăţi:

$$\begin{gathered} \forall n \in \mathbb{N}: \quad \omega > n, \\ \forall n \in \mathbb{N}: \quad \omega + n = \omega, \\ \omega \geq \omega. \end{gathered} \tag{BT3.3.1}$$

Construcţia sistematică a arborelui de acoperire a unei reţele ($\boldsymbol{N}$, M_0) se desfăşoară conform următorului algoritm (ilustrare în **AP3.3**):

Pas 1. Se stabileşte M_0 ca rădăcină şi se etichetează M_0 ca "*marcaj nou*".

Pas 2. Atât timp cât există cel puţin un marcaj etichetat drept "*marcaj nou*" se efectuează următorii subpaşi:

Subpas 2.1. Se selectează un "*marcaj nou*" *M*.

Subpas 2.2. Dacă *M* este identic cu un marcaj de pe drumul de la rădăcină la *M*, atunci marcajul *M* se etichetează drept "*marcaj vechi*" şi se trece la un alt "*marcaj nou*".

Subpas 2.3. Dacă pentru *M*, nici o tranziţie nu este validată, atunci *M* se etichetează ca "*marcaj de deadlock*".

Subpas 2.4. Dacă pentru *M* există tranziţii validate, atunci pentru *fiecare* tranziţie *t* validată se efectuează următoarele etape:

Etapa 2.4.1. Se obţine marcajul *M′* care rezultă din executarea tranziţiei *t*, pornind de la marcajul *M.*

Etapa 2.4.2. Dacă pe drumul de la rădăcină la *M* există un marcaj *M″* astfel încât $M'(p) \geq M''(p)$ pentru orice poziţie *p* şi $M' \neq M''$ (în sensul că pentru cel puţin o poziţie *p* inegalitatea $M'(p) \geq M''(p)$ este strictă), atunci $M'(p)$ se înlocuieşte cu ω pentru fiecare poziţie *p* în care avem inegalitatea strictă $M'(p) > M''(p)$ (adică marcajul *M′* acoperă marcajul *M″*.)

Etapa 2.4.3. Se introduce *M′* ca nod al arborelui de acoperire, se trasează un arc de la *M* la *M′* corespunzând tranziţiei *t* şi se etichetează *M′* drept "*marcaj nou*".

Observaţie: În cazul reţelelor *nemărginite*, studierea proprietăţii de *viabilitate* cu ajutorul arborelui de acoperire nu este întotdeauna posibilă. Acest fapt se datorează pierderii (prin utilizarea simbolului ω) unor informaţii numerice concrete, referitoare la marcajul poziţiilor nemărginite (de exemplu creşterea sau descreşterea marcajului).

BT3.3.2. Grafuri de acoperire/accesibilitate

Graful de acoperire (eng. *coverability graph*) asociat unei reţele Petri ($\boldsymbol{N}$, M_0) mărginite, este un graf orientat $G = (V, E)$. Mulţimea nodurilor *V* este dată de mulţimea tuturor marcajelor distincte din arborele de acoperire. Mulţimea arcelor orientate *E* serveşte pentru a uni oricare două marcaje M_i, M_j din *V*, dacă există o tranziţie t_k a cărei executare duce de la M_i la M_j; arcele din *E* corespund arcelor din arborele de acoperire (ilustrare în **AP3.3**).

În cazul reţelelor Petri *mărginite*, graful de acoperire este referit drept *graf de accesibilitate* (eng. *reachability graph*), întrucât nodurile acestuia sunt chiar marcajele accesibile ale reţelei, adică $V \equiv R(M_0)$. Grafurile de accesibilitate pot fi utilizate pentru studierea tuturor proprietăţilor comportamentale ale reţelelor Petri mărginite (ilustrare în **AP3.1** şi **AP3.2**).

BT3.4. Ecuaţia de stare

Se consideră o reţea Petri pură $\boldsymbol{N}$ (în care nu există bucle autonome), cu *n* tranziţii şi *m* poziţii. Se numeşte *matrice de incidenţă* (eng. *incidence matrix*) a reţelei, o matrice $\boldsymbol{A} = [a_{ij}]$ de dimensiune $n \times m$, ale cărei elemente sunt numere întregi:

$$a_{ij} = a_{ij}^{+} - a_{ij}^{-}; \quad i = 1,...,n, \quad j = 1,...,m, \tag{BT3.4.1}$$

unde:

- $a_{ij}^{+} = W(t_i, p_j)$ este ponderea arcului *de la* tranziţia t_i, *către* poziţia sa de ieşire p_j;
- $a_{ij}^{-} = W(p_j, t_i)$ este ponderea arcului *către* tranziţia t_i, *de la* poziţia sa de intrare p_j.

Matricele $\boldsymbol{A}^{+} = [a_{ij}^{+}]$ şi $\boldsymbol{A}^{-} = [a_{ij}^{-}]$ (de dimensiune $n \times m$) sunt referite drept *matrice de incidenţă de ieşire*, respectiv *matrice de incidenţă de intrare*. Din punctul de vedere al aplicaţiilor, scrierea matricei de incidenţă $\boldsymbol{A}$ se poate face global, construind, mai întâi, matricele $\boldsymbol{A}^{+}$ şi $\boldsymbol{A}^{-}$, după care se efectuează diferenţa $\boldsymbol{A} = \boldsymbol{A}^{+} - \boldsymbol{A}^{-}$.

Conform regulii de validare şi executare a tranziţiei, se observă că a_{ij}^{-} şi a_{ij}^{+} reprezintă numărul de jetoane îndepărtate şi, respectiv, adăugate în poziţia p_j, atunci când tranziţia t_i se execută o singură dată.

Să considerăm o secvenţă de executări de tranziţii şi să presupunem că cea de a k-a executare din această secvenţă are loc în tranziţia t_i, adică tranziţia desemnată prin t_i se află pe locul k în secvenţa de executări $\sigma = \underbrace{t_a}_{\text{locul 1}} \; \underbrace{t_b}_{\text{locul 2}} \; \ldots \; \underbrace{t_i}_{\text{locul } k} \; \ldots \; \underbrace{t_d}_{\text{locul } q}$.

Considerăm vectorul coloană $\boldsymbol{u}_k$ (de dimensiune $n \times 1$) ale cărui elemente sunt toate 0, cu excepţia celui de al i-lea element care este 1 (corespunzător tranziţiei t_i, unde are loc cea de a k-a executare a secvenţei) şi notăm cu M_{k-1}, M_k vectorii coloană (de dimensiune $m \times 1$) corespunzători marcajului rezultat după cea de-a (k–1)-a, respectiv a k-a executare din secvenţa considerată. Schimbarea de marcaj după cea de a k-a executare este descrisă de egalitatea:

$$M_k - M_{k-1} = \boldsymbol{A}^T \boldsymbol{u}_k, \quad k = 1,2,\ldots. \tag{BT3.4.2}$$

Această egalitate este uzual folosită sub forma echivalentă:

$$M_k = M_{k-1} + \boldsymbol{A}^T \boldsymbol{u}_k, \quad k = 1,2,\ldots \tag{BT3.4.3}$$

şi reprezintă *ecuaţia de stare* (eng. *state equation*) a reţelei Petri. Vectorul $\boldsymbol{u}_k$ se numeşte *vector de executare* sau *vector de control* (eng. *control vector*) (ilustrare în **AP3.4**).

Scriind ecuaţia de stare (BT3.4.3) pentru $k = 1,2,...,q$ şi sumând, se obţine marcajul la care se ajunge pornind din M_0, după executarea secvenţei de tranziţii σ:

$$M_q = M_0 + \boldsymbol{A}^T \sum_{k=1}^{q} \boldsymbol{u}_k \,. \tag{BT3.4.4}$$

Ecuaţia de stare poate fi utilizată pentru a aborda *probleme de accesibilitate*. Dacă un marcaj destinaţie M_d este accesibil din M_0 prin secvenţa de executări descrisă de vectorii $\boldsymbol{u}_1$, $\boldsymbol{u}_2$,..., $\boldsymbol{u}_d$, ecuaţia (BT3.4.4) conduce la:

$$\boldsymbol{A}^T \boldsymbol{x} = \Delta M \,, \tag{BT3.4.5}$$

unde $\Delta M = M_d - M_0$ reprezintă diferenţa de marcaj şi vectorul coloană $\boldsymbol{x} = \sum_{k=1}^{d} \boldsymbol{u}_k$ (care are toate elementele întregi nenegative) se numeşte *vectorul numărului de executări posibile*. Cel

de-al i-lea element al vectorului $\boldsymbol{x}$, $i=1,...,n$, conţine numărul de executări ale tranziţiei t_i, $i=1,...,n$, în secvenţa ce transferă M_0 în M_d. Ecuaţia (BT3.4.5) furnizează o *condiţie de tip necesar pentru accesibilitate*.

Teorema 3.4.1. Dacă pentru reţeaua Petri ($\boldsymbol{N}$, M_0), marcajul M_d este accesibil din M_0, atunci are loc egalitatea:

$$\text{rang}\, \boldsymbol{A}^T = \text{rang}\, [\boldsymbol{A}^T \vdots \Delta M]\,, \qquad \text{(BT3.4.6)}$$

unde $\boldsymbol{A}$ notează matricea de incidenţă a reţelei, iar ΔM este diferenţa de marcaj.

Importanţa practică a **Teoremei 3.4.1** este dată de forma *echivalentă*, exprimată prin contrara reciprocei, care oferă o *condiţie suficientă pentru neaccesibilitate*.

Teorema 3.4.2. Un marcaj M_d nu este accesibil din M_0, dacă egalitatea de ranguri din (BT3.4.6) nu este satisfăcută (ilustrare în **AP3.4**).

BT3.5. Proprietăţi ce decurg din apartenenţa reţelelor la anumite subclase de reţele Petri ordinare

În paragraful BT2.5 au fost prezentate sumar clasele de reţele Petri ordinare frecvent întâlnite în modelarea sistemelor cu evenimente discrete. Pentru două dintre acestea, şi anume maşinile de stare şi grafurile marcate, sunt prezentate în continuare metode de analiză a proprietăţilor de *viabilitate* şi *siguranţă*. Pentru celelalte clase de reţele Petri ordinare, vezi (Păstrăvanu, 1997).

Teorema 3.5.1. Dacă reţeaua Petri ($\boldsymbol{N}$, M_0) este viabilă şi sigură, atunci nu conţine nici un nod (tranziţie sau poziţie) de tip sursă sau receptor, adică:

$$\forall x \in P \bigcup T: \quad {}^{\bullet}x \neq \varnothing \;\text{ şi }\; x^{\bullet} \neq \varnothing\,. \qquad \text{(BT3.5.1)}$$

Această *condiţie necesară* este utilizată în practică în forma echivalentă (contrara reciprocei) care reprezintă un *criteriu suficient* pentru inexistenţa proprietăţilor de viabilitate *sau* siguranţă.

Teorema 3.5.2. Dacă relaţia (BT3.5.1) nu este satisfăcută, atunci reţeaua Petri ($\boldsymbol{N}$, M_0) este fie neviabilă, fie nesigură, fie neviabilă şi nesigură.

O extindere a **Teoremei 3.5.1** o reprezintă următoarea teoremă.

Teorema 3.5.3. Dacă reţeaua Petri ($\boldsymbol{N}$, M_0) conexă (adică există un drum neorientat de la orice nod către orice nod din $P \bigcup T$). este viabilă şi sigură, atunci $\boldsymbol{N}$ este un graf *tare conex* (adică există un drum orientat de la orice nod către orice nod din $P \bigcup T$).

BT3.5.1. Criterii de viabilitate şi siguranţă a maşinilor de stare

Pentru *maşinile de stare* există următoarele condiţii *necesare şi suficiente* care decurg din criteriul de necesitate 3.5.3 şi din faptul că executarea unei tranziţii, în acest tip de reţea, deplasează *un singur* jeton dintr-o poziţie în altă poziţie.

Teorema 3.5.4. O maşină de stare ($\boldsymbol{N}$, M_0) este viabilă dacă şi numai dacă $\boldsymbol{N}$ este tare conex şi M_0 are cel puţin un jeton.

Teorema 3.5.5. O maşină de stare ($\boldsymbol{N}$, M_0) este sigură dacă şi numai dacă M_0 are cel mult un jeton.

Teorema 3.5.6. O maşină de stare ($\boldsymbol{N}$, M_0) viabilă este sigură dacă şi numai dacă M_0 are un singur jeton.

BT3.5.2. Criterii de viabilitate şi siguranţă a grafurilor marcate

În fiecare poziţie a unui graf marcat intră şi, respectiv, iese câte un singur arc. Executarea unei tranziţii constă în îndepărtarea unui singur jeton din fiecare poziţie de intrare şi adăugarea unui singur jeton în fiecare poziţie de ieşire a tranziţiei în cauză.

Din topologia grafului marcat, constatăm că dacă o tranziţie este plasată pe un anumit arc orientat, atunci o *singură* poziţie de intrare şi o *singură* poziţie de ieşire a acestei tranziţii aparţin *circuitului orientat* considerat. Din această constatare decurge următoarea *proprietate de invarianţă*.

Teorema 3.5.7. Într-un graf marcat, numărul de jetoane în orice circuit orientat este invariant în raport cu numărul de executări, adică, pentru orice marcaj $M \in R(M_0)$ şi pentru orice circuit C, avem egalitatea:

$$M(C) = M_0(C)\,. \qquad \text{(BT3.5.2)}$$

O consecinţă imediată a acestui rezultat îl constituie faptul că dacă marcajul iniţial nu plasează nici un jeton pe un anume circuit orientat, atunci, pentru orice marcaj accesibil, acest circuit va rămâne fără nici un jeton. Cu alte cuvinte, tranziţiile acestui circuit nu vor putea fi validate niciodată.

Să privim acum lucrurile în sens invers şi să presupunem că o anumită tranziţie nu ajunge să fie validată niciodată. Invalidarea este consecinţa existenţei unei poziţii de intrare fără nici un jeton. Pornind de la această poziţie şi deplasându-ne înapoi (pe arcele reţelei), vom ajunge, în final, la un circuit orientat care nu conţine nici un jeton. Prin acest raţionament, am demonstrat, de fapt, următoarea *condiţie necesară şi suficientă de viabilitate*.

Teorema 3.5.8. Un graf marcat este viabil, dacă şi numai dacă marcajul iniţial M_0 plasează *cel puţin* un jeton pe fiecare circuit orientat (ilustrare în **AP3.4**).

Întrucât o poziţie poate aparţine mai multor circuite, furnizăm încă un rezultat referitor la proprietăţile marcajului.

Teorema 3.5.9. Numărul *maxim* de jetoane pe care le poate conţine o poziţie p a grafului marcat este egal cu *minimul* dintre numărul de jetoane pe care le plasează marcajul

iniţial M_0 pe circuitele orientate C_1, C_2,..., C_q, care conţin poziţia respectivă:

$$M(p) \le \min \{M_0(C_1),..., M_0(C_q)\}. \qquad \text{(BT3.5.3)}$$

Acum suntem în măsură să formulăm o condiţie *necesară şi suficientă de siguranţă*.

Teorema 3.5.10. Un graf marcat viabil este sigur dacă şi numai dacă *orice* poziţie aparţine unui circuit orientat C, cu marcajul iniţial $M_0(C) = 1$ (ilustrare în **AP3.4**).

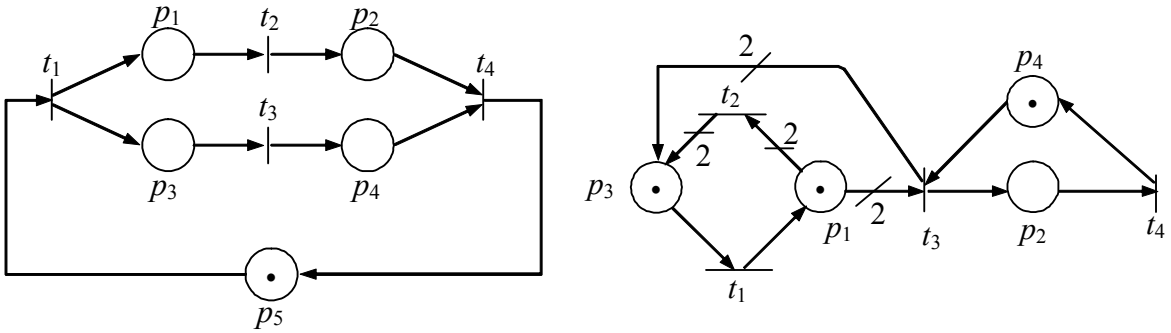
În final putem intra în posesia unei condiţii *necesare şi suficiente de existenţă* a unui marcaj iniţial *viabil şi sigur*.

Teorema 3.5.11. Într-un graf marcat există un marcaj iniţial viabil şi sigur dacă şi numai dacă acesta este tare conex.

Aplicaţii

AP3.1.

Se consideră reţelele Petri din fig. AP3.1.1. Pentru fiecare dintre aceste reţele, să se construiască arborele şi graful de accesibilitate.



(a) (b)

Fig. AP3.1.1. *Reţelele Petri utilizate în aplicaţia AP3.1.*

<u>*Soluţie*</u>

A. Arborele de accesibilitate asociat reţelei Petri din fig. AP3.1.1.(a) este prezentat în fig. AP3.1.2.(a). Pentru a reprezenta marcajele etichetate drept "*marcaj vechi*" s-au utilizat caractere aldine. Graful de accesibilitate corespunzător este prezentat în fig. AP3.1.2.(b).

B. Arborele de accesibilitate asociat reţelei Petri din fig. AP3.1.1.(b) este prezentat în fig. AP3.1.3.(a). Pentru a reprezenta marcajele etichetate drept "*marcaj vechi*" s-au utilizat caractere aldine. Graful de accesibilitate corespunzător este prezentat în fig. AP3.1.3.(b).

Pornind de la informaţiile din această soluţie, cititorul este invitat să studieze proprietăţile comportamentale ale celor două reţele din fig. AP3.1.1.

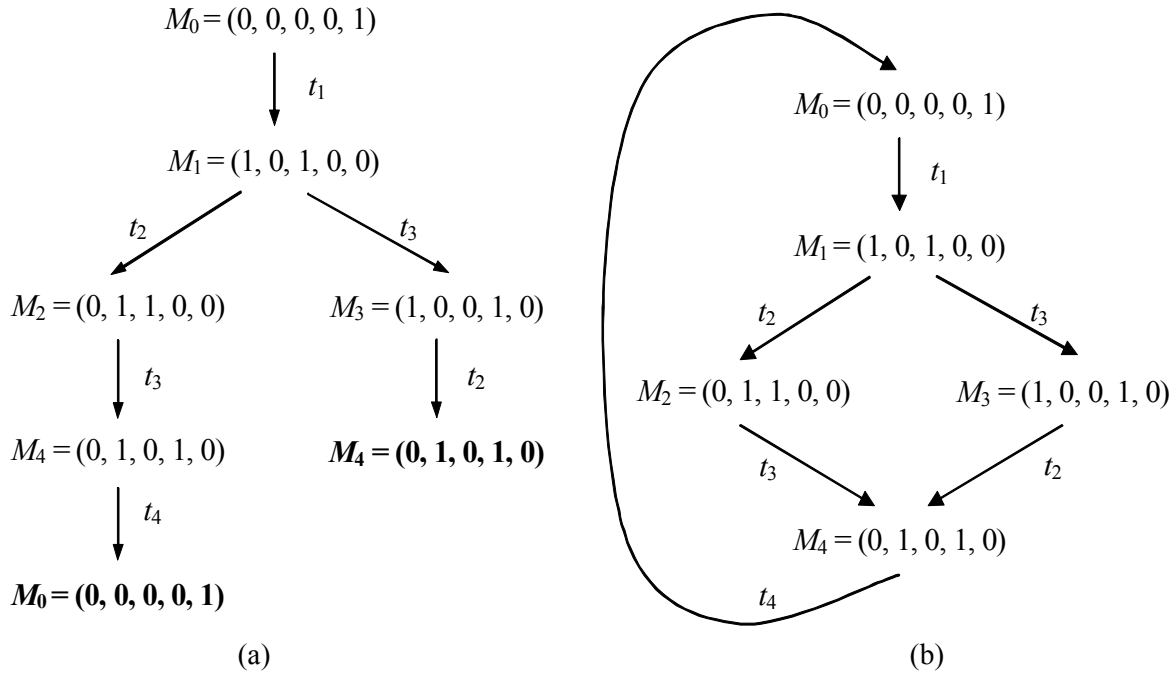


Fig. AP3.1.2. *Arborele* (a) *şi graful de accesibilitate* (b) *pentru reţeaua din fig. AP3.1.1.*(*a*).

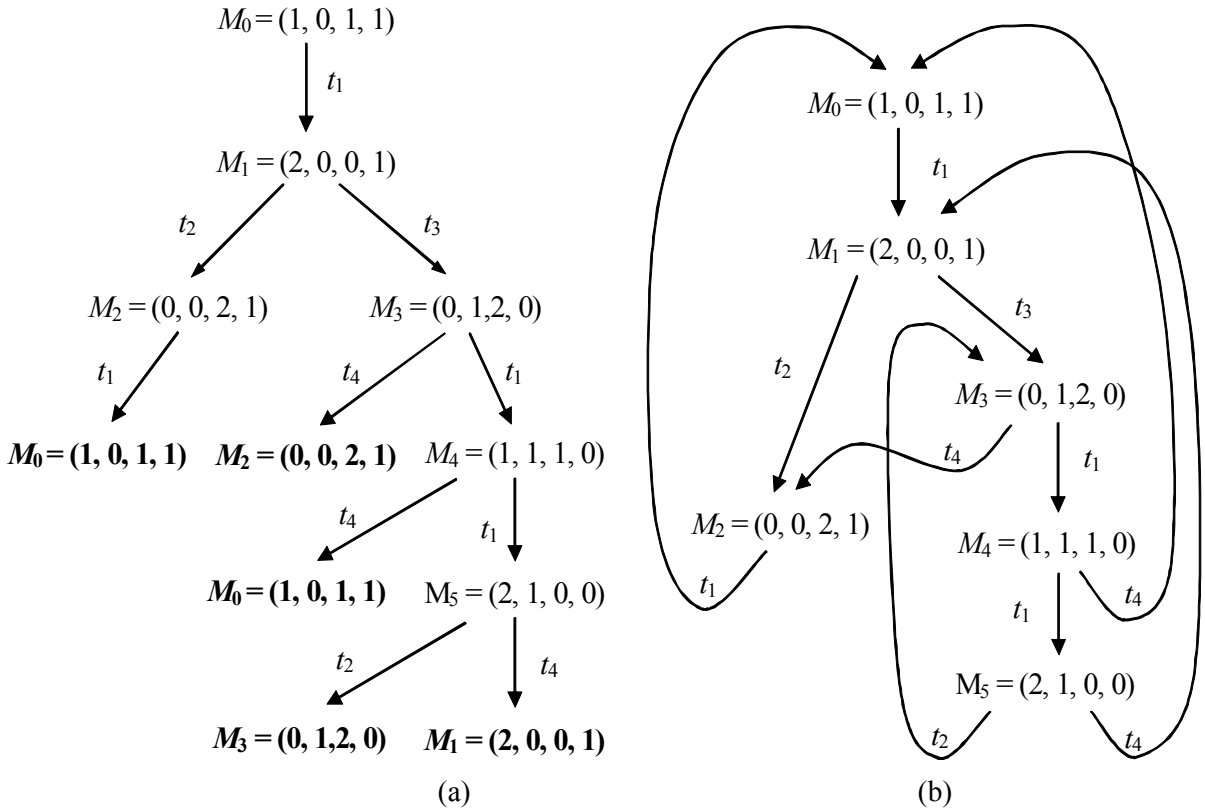


Fig. AP3.1.3. *Arborele* (a) *şi graful de accesibilitate* (b) *pentru reţeaua din fig. AP3.1.1.*(*b*).

AP3.2.

Să se construiască arborele şi graful de accesibilitate pentru reţeaua Petri din fig. AP3.2.1.

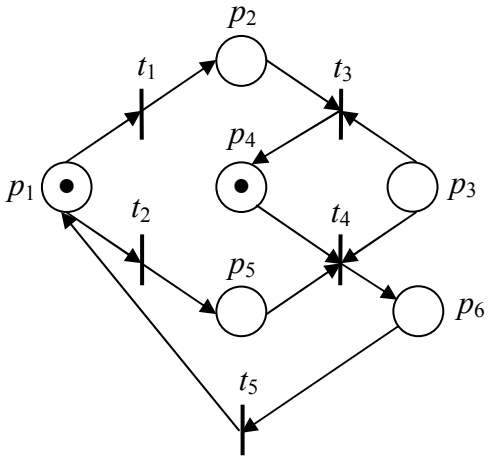


Fig. AP3.2.1. *Reţeaua Petri utilizată în aplicaţia AP3.2.*

Soluţie

Arborele de accesibilitate al reţelei Petri din fig. AP3.2.1 este prezentat în fig. AP3.2.2. Marcajele de deadlock sunt evidenţiate prin încadrarea într-un chenar. Se observă că, pentru această reţea Petri, arborele şi graful de accesibilitate coincid.

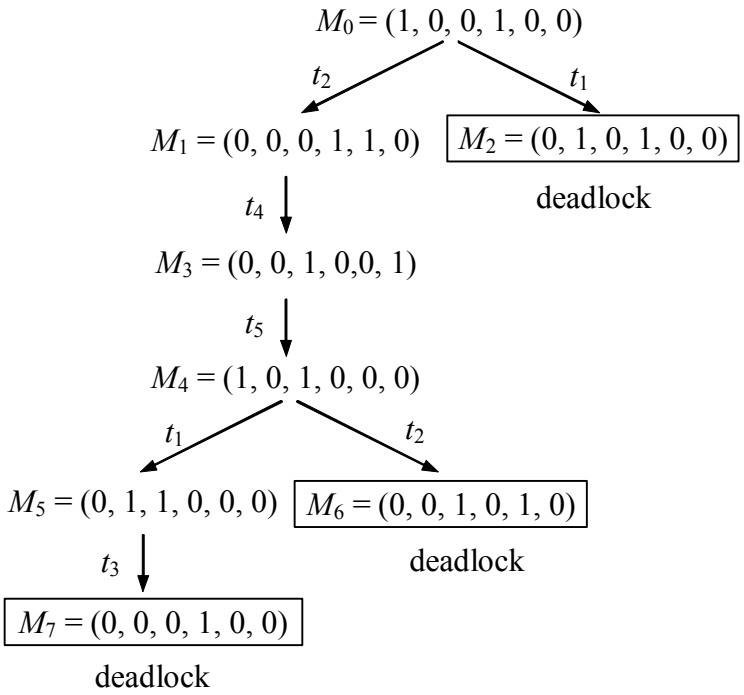
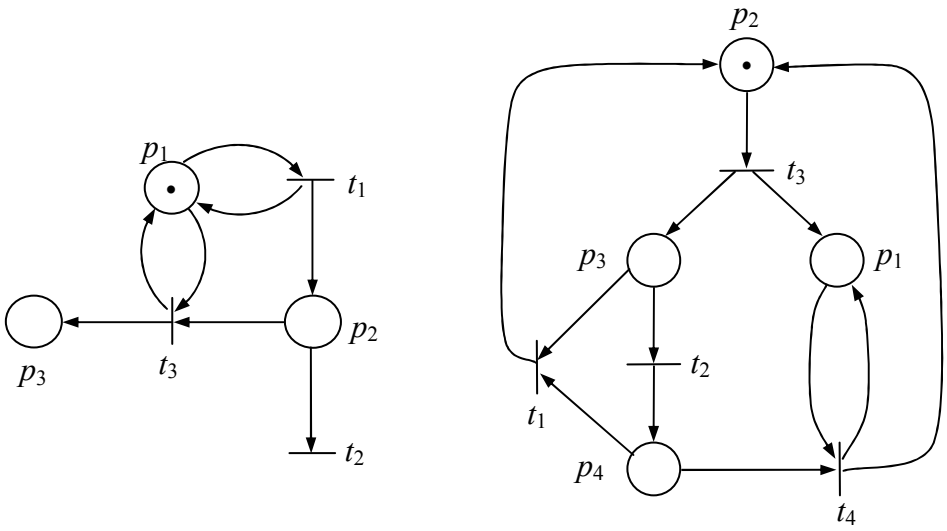


Fig. AP3.2.2. *Arborele* (*graful*) *de accesibilitate al reţelei Petri din fig. AP3.2.1.*

Pornind de la informaţiile din această soluţie, cititorul este invitat să studieze proprietăţile comportamentale ale reţelei din fig. AP3.2.1.

AP3.3.

Să se construiască arborele şi graful de acoperire pentru reţelele Petri de capacitate infinită din fig. AP3.3.1.

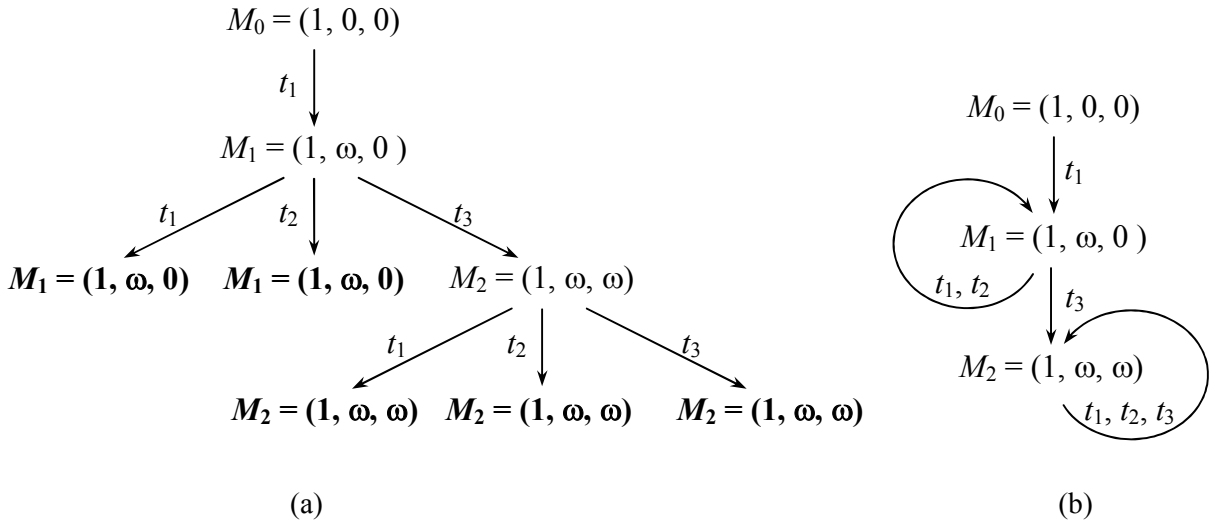


(a) (b)

Fig. AP3.3.1. *Reţeaua Petri utilizată în aplicaţia AP3.3.*

Soluţie

A. Arborele de acoperire asociat reţelei Petri din fig. AP3.3.1.(a) este prezentat în fig. AP3.3.2.(a). Pentru a reprezenta marcajele etichetate drept "*marcaj vechi*" s-au utilizat caractere aldine.



(a) (b)

Fig. AP3.3.2. *Arborele* (a) *şi graful de accesibilitate* (b) *pentru reţeaua din fig. AP3.3.1.*(*a*).

B. Arborele de acoperire asociat reţelei Petri din fig. AP3.3.1.(a) este prezentat în fig. AP3.3.2.(a). Pentru a reprezenta marcajele etichetate drept "*marcaj vechi*" s-au utilizat caractere aldine.

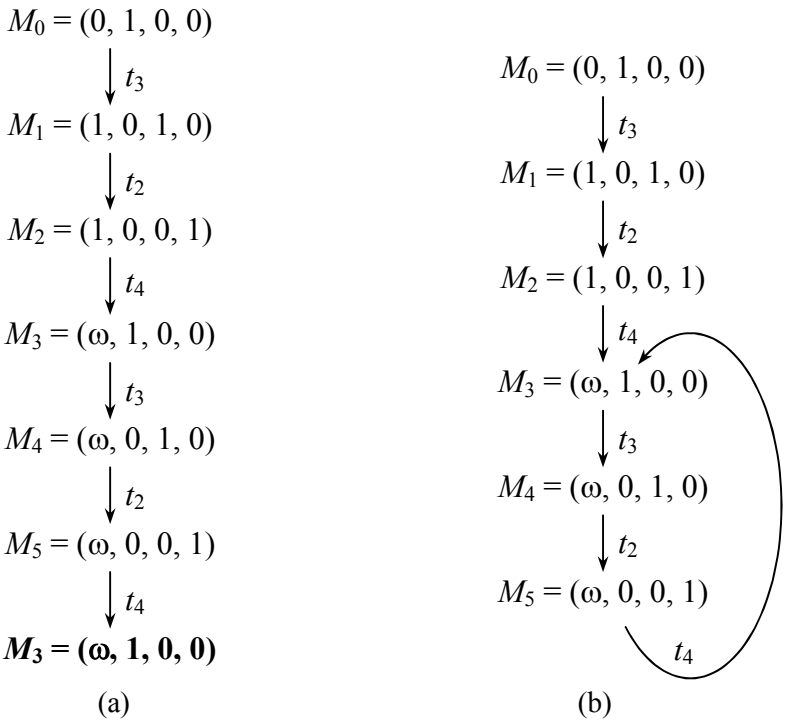


Fig. AP3.3.3. *Arborele* (a) *şi graful de accesibilitate* (b) *pentru reţeaua din fig. AP3.3.1.*(*b*).

Pornind de la informaţiile din această soluţie, cititorul este invitat să studieze proprietăţile comportamentale ale celor două reţele din fig. AP3.3.1.

AP3.4.

Se consideră protocolul de comunicaţii studiat în **AP2.4**.

1. Să se verifice că reţeaua Petri care îl modelează (fig. AP2.4.1) are proprietăţile de viabilitate, siguranţă şi reversibilitate.
2. Să se scrie ecuaţia de stare a reţelei Petri.
3. Utilizând ecuaţia de stare, să se arate că marcajul $M = [0\ 0\ 0\ 1\ 1\ 0\ 1\ 0]^T$ nu este accesibil din M_0. Să se formuleze o interpretare fizică a acestui rezultat.

Soluţie

1. Studiind arborele de accesibilitate al reţelei Petri din fig. AP2.4.1, prezentat în fig. AP2.4.2, putem afirma că reţeaua este viabilă, sigură (1-mărginită), şi reversibilă.

Deoarece reţeaua Petri studiată este de tip graf marcat, pentru studiul proprietăţilor de viabilitate şi siguranţă se pot aplica **Teoremele 3.5.8** şi **3.5.10**. Fiecare poziţie a reţelei aparţine unuia dintre următoarele trei circuite elementare orientate $C_1 = p_1 t_1 p_4 t_6 p_6 t_3$, $C_2 = p_1 t_1 p_2 t_2 p_5 t_4 p_7 t_6 p_6 t_3$, $C_3 = p_3 t_2 p_5 t_4 p_8 t_5$. Deoarece fiecare dintre aceste circuite are câte un singur jeton în marcajul iniţial, rezultă că reţeaua este viabilă şi sigură.

2. Matricea de incidenţă de ieşire şi, respectiv, matricea de incidenţă de intrare, sunt date de:

$$\boldsymbol{A}^+ = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{A}^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \boldsymbol{A}^+, \boldsymbol{A}^- \in \mathbb{Z}^{6\times 8},$$

astfel că matricea de incidenţă este:

$$\boldsymbol{A} = \boldsymbol{A}^+ - \boldsymbol{A}^- = \begin{bmatrix} -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 1 & -1 & 0 \end{bmatrix}, \quad \boldsymbol{A} \in \mathbb{Z}^{6\times 8}.$$

Să considerăm o secvenţă de executări de tranziţii şi să presupunem că cea de a *k*-a executare din această secvenţă are loc în tranziţia t_i, adică tranziţia desemnată prin t_i se află pe locul *k* în secvenţa de executări $\sigma = \underbrace{t_a}_{\text{locul 1}} \; \underbrace{t_b}_{\text{locul 2}} \; \ldots \; \underbrace{t_i}_{\text{locul } k} \; \ldots \; \underbrace{t_d}_{\text{locul } q}$.

Ecuaţia de stare a reţelei Petri este:

$$M_k = M_{k-1} + \boldsymbol{A}^T \boldsymbol{u}_k, \quad k = 1, 2, \ldots,$$

unde $\boldsymbol{u}_k \in \mathbb{N}^6$ este vectorul de executare, ale cărui elemente sunt toate 0, cu excepţia celui de al *i*-lea element care este 1 (corespunzător tranziţiei t_i, unde are loc cea de a *k*-a executare din secvenţa σ). Vectorii coloană $M_{k-1}, M_k \in \mathbb{N}^8$ corespund marcajului rezultat după cea de-a (*k*–1)-a şi, respectiv, cea de-a a *k*-a executare din secvenţa considerată.

3. Pentru a verifica dacă marcajul *M* este accesibil din M_0 se aplică **Teorema 3.4.2** (care reprezintă un criteriu de incompatibilitate a ecuaţiei algebrice (BT3.4.5)). În acest scop construim vectorul $\Delta M = M - M_0 = \begin{bmatrix} -1 & 0 & -1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}^T$. Deoarece $\operatorname{rang} \boldsymbol{A}^T = 5$ şi $\operatorname{rang} [\boldsymbol{A}^T \vdots \Delta M] = 6$, rezultă că marcajul *M* nu este accesibil din M_0.

AP3.5.

Se consideră un sistem de calcul biprocesor, echipat cu două unităţi de disc D_1, D_2. Fiecare din cele două procesoare P_1, P_2 execută câte o succesiune de taskuri, fiecare task necesitând ambele discuri. Succesiunea de taskuri executate de P_1 este referită prin ST_1, iar succesiunea de taskuri executată de P_2 este referită prin ST_2. Execuţia fiecărui task din ST_1 necesită mai întâi D_1; apoi, păstrând alocarea lui D_1, se solicită alocarea lui D_2; în final D_1 şi D_2 sunt

eliberate simultan. Execuţia fiecărui task din ST_2 necesită mai întâi D_2; apoi, păstrând alocarea lui D_2, se solicită alocarea lui D_1; în final D_1 şi D_2 sunt eliberate simultan. O reprezentare schematizată a funcţionării sistemului de calcul este dată în fig. AP3.3.1.

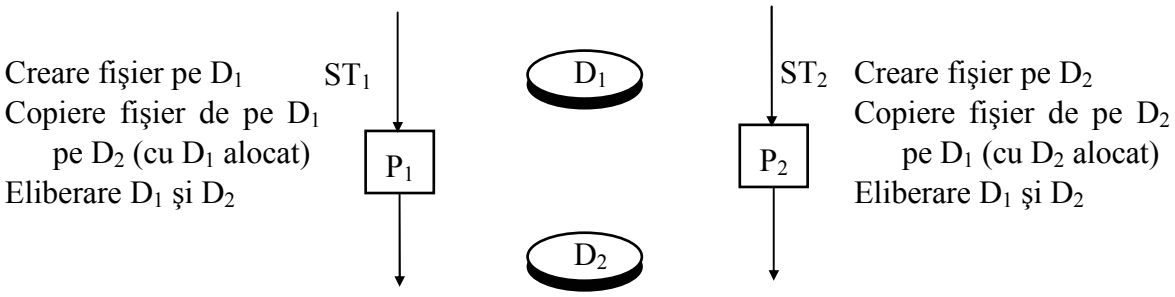


Fig. AP3.5.1. *Reprezentarea schematizată a funcţionării sistemului de calcul studiat în AP3.5.*

Nu există nici o informaţie apriori privind duratele de timp cât taskurile din ST_1 sau ST_2 ocupă D_1 şi D_2. La momentul iniţial atât procesoarele, cât şi discurile, sunt libere.

1. Să se construiască un model tip reţea Petri netemporizată pentru funcţionarea sistemului de calcul, precizând semnificaţia fizică a poziţiilor şi a tranziţiilor.
2. Să se construiască graful de accesibilitate pentru modelul de tip reţea Petri.
3. Să se investigheze posibilitatea producerii fenomenului de deadlock în sistem.
   a) Se va preciza marcajul reţelei Petri pentru care se ajunge în deadlock precum şi secvenţele de executări de tranziţii care conduc la acesta.
   b) Se va preciza succesiunea de operaţii executate în sistemul de calcul care conduce la deadlock şi se va explica modul de blocare circulară a resurselor.
4. Să se modifice modelul construit la punctul 1 astfel încât să se poată evita producerea fenomenului de deadlock.
5. Să se simuleze în mediul ***Petri Net Toolbox*** modelul obţinut la punctul 4, pentru 10.000 de taskuri servite complet de sistem, în următoarele situaţii de funcţionare a sistemului:
   (i) Taskurile din ST_1 şi cele din ST_2 au aceeaşi probabilitate de servire.
   (ii) Taskurile din ST_1 au probabilitate de servire mult mai mare decât cele din ST_2.
   (iii) Taskurile din ST_2 au probabilitate de servire mult mai mare decât cele din ST_1.

   Pentru fiecare din cele trei situaţii se va preciza numărul total de taskuri servite complet de fiecare din cele două procesoare.

Soluţie

1. Modelul de tip reţea Petri corespunzător sistemului de calcul este prezentat în fig. AP3.5.2. Semnificaţia fizică a poziţiilor şi tranziţiilor este următoarea:

- pentru poziţii: P1 – procesor P_1 disponibil; P1D1 – creare fişier pe D_1; P1D1D2 – copiere fişier de pe D_1 pe D_2; P2 – procesor P_2 disponibil; P2D2 – creare fişier pe D_2; P2D2D1 – copiere fişier de pe D_2 pe D_1; D1 – disc D_1 disponibil; D2 – disc D_2 disponibil;
- pentru tranziţii: t1 – începerea operaţiei de creare fişier pe discul D_1 de către procesorul P_1; t2 – terminarea creării fişierului pe D_1 şi începerea copierii pe discul D_2

de către P_1; t3 – terminarea unui task din ST_1; t4 – începerea operaţiei de creare fişier pe discul D_2 de către procesorul P_2; t5 – terminarea creării fişierului pe D_2 şi începerea copierii pe discul D_1 de către P_2; t6 – terminarea unui task din ST_2.

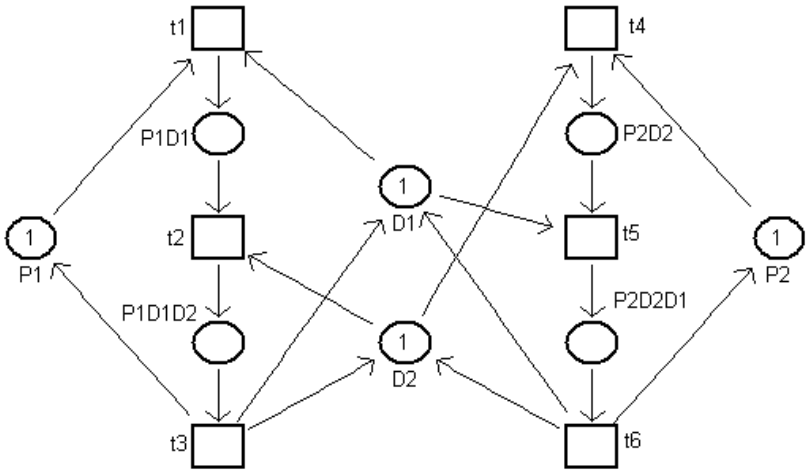


Fig. AP3.5.2. *Modelul tip reţea Petri al sistemului de calcul biprocesor din fig. AP3.3.1.*

Observaţie: Reţeaua Petri din fig. AP3.5.2 constituie un model pentru binecunoscuta problemă teoretică a doi filozofi chinezi care cinează împreună, formulată în (Dijkstra, 1968).

2. Arborele de accesibilitate corespunzător reţelei Petri din fig. AP3.5.2, construit de către mediul ***Petri Net Toolbox*** pe baza topologiei şi marcajului iniţial al reţelei, este prezentat în fig. AP3.5.3.

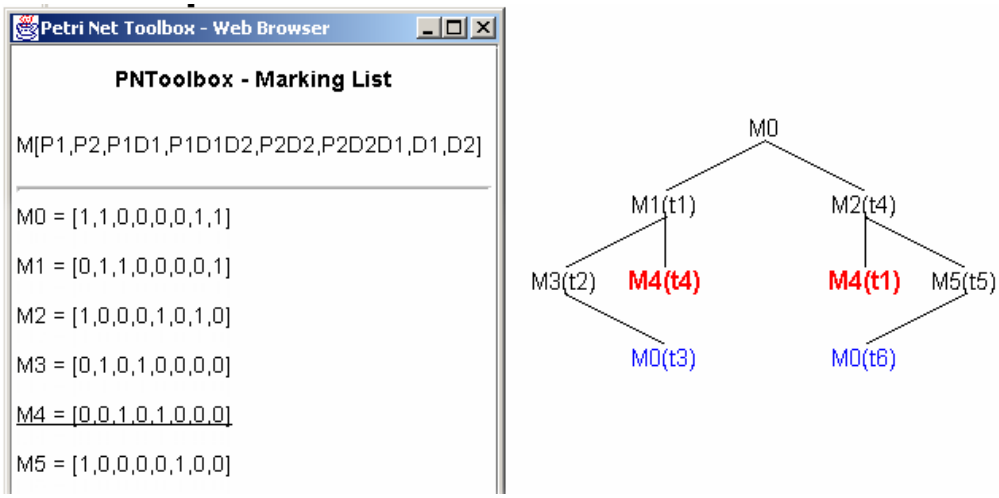


Fig. AP3.5.3. *Arborele de acoperire al reţelei Petri din fig. AP3.3.2.*

3. Pe arborele de accesibilitate prezentat în fig. AP3.5.3 se observă existenţa marcajului de deadlock M_4 = (0, 0, 1, 0, 1, 0, 0, 0). Secvenţele de executări de tranziţii care conduc din marcajul iniţial la cel de deadlock sunt $\sigma_1 = t_1 t_4$ şi $\sigma_2 = t_4 t_1$.

Secvenţa σ_1 corespunde următoarei succesiuni de operaţii care are loc în sistemul de calcul: Procesorului P_1 i se alocă discul D_1, după care discul D_2 este alocat procesorului P_2. Succesiunea de operaţii corespunzătoare secvenţei σ_2 este: Procesorului P_2 i se alocă discul D_2, după care discul D_1 este alocat procesorului P_1. Se ajunge astfel în starea în care fiecare disc este alocat pentru prima operaţie efectuată de câte unul dintre procesoare şi nu poate fi eliberat deoarece fiecare procesor are nevoie (pentru terminarea corectă a taskului început) şi de cel de-al doilea disc.

Datorită faptului că cele două discuri reprezintă resurse partajate în paralel de ST_1 şi ST_2 şi nu există nici un mecanism de asigurare a excluderii mutuale paralele, se ajunge la blocajul circular de resurse în sistemul de calcul.

4. Pentru rezolvarea situaţiilor conflictuale şi evitarea fenomenul de deadlock se pot utiliza mai multe metode care au în vedere impunerea unor condiţii astfel încât un task odată început să-şi termine execuţia.

Soluţia 1: Introducem o poziţie suplimentară (notată cu V în fig. AP3.5.4.) care validează începerea unui task ce va fi complet servit.

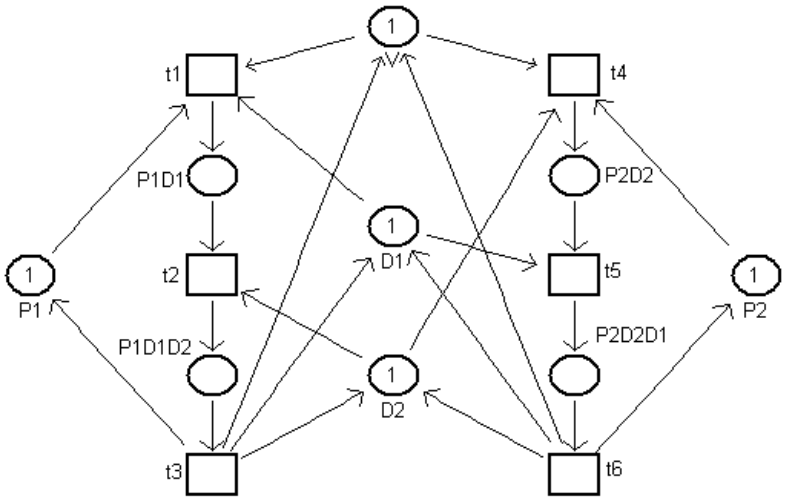


Fig. AP3.5.4. *Modelul de tip reţea Petri al sistemului de calcul prezentat în fig. AP3.5.2 cu o poziţie suplimentară de validare.*

Soluţia 2: Dacă consultăm arborele de acoperire al reţelei Petri reprezentate în fig. AP3.5.2, observăm că în marcajul $M_1 = (0, 1, 1, 0, 0, 0, 0, 1)$ sunt validate tranziţiile t_2 şi t_4. Executarea tranziţiei t_2 asigură terminarea servirii unui client de către procesorul P_1 şi eliberarea resurselor, în timp ce executarea lui t_4 conduce la deadlock. Similar, în marcajul $M_2 = (1, 0, 0, 0, 1, 0, 1, 0)$ sunt validate tranziţiile t_1 şi t_5, executarea lui t_1 conducând la deadlock.

Fenomenul de deadlock poate fi evitat impunând priorităţi între tranziţiile în conflict, şi anume executarea tranziţiei t_2 să aibă prioritate faţă de executarea tranziţiei t_4 şi, similar, executarea tranziţiei t_5 să aibă prioritate faţă de executarea tranziţiei t_1.

Soluţia 3: O altă metodă ce poate fi utilizată pentru a asigura evitarea blocajului circular de resurse este de a inhiba executarea tranziţiei t_4 după executarea lui t_1, adică atunci când discul

D_1 este alocat procesorului P_1. În acelaşi timp, se inhibă executarea tranziţiei t_1 după executarea lui t_4, adică atunci când discul D_2 este alocat procesorului P_2 (fig. AP3.5.5).

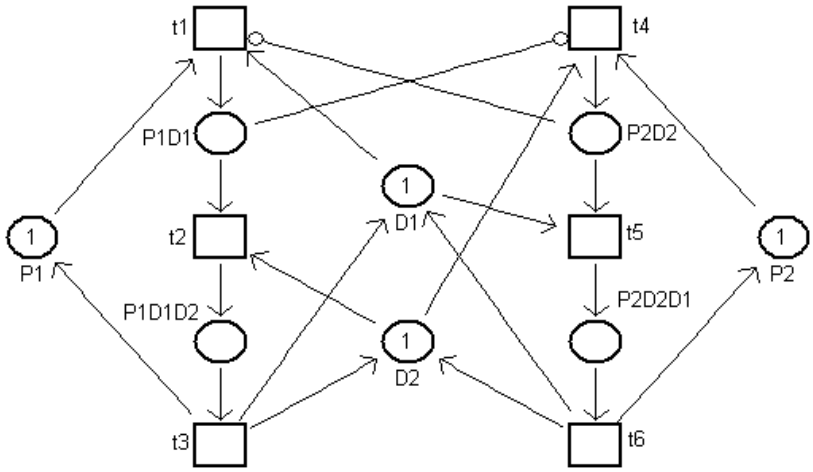
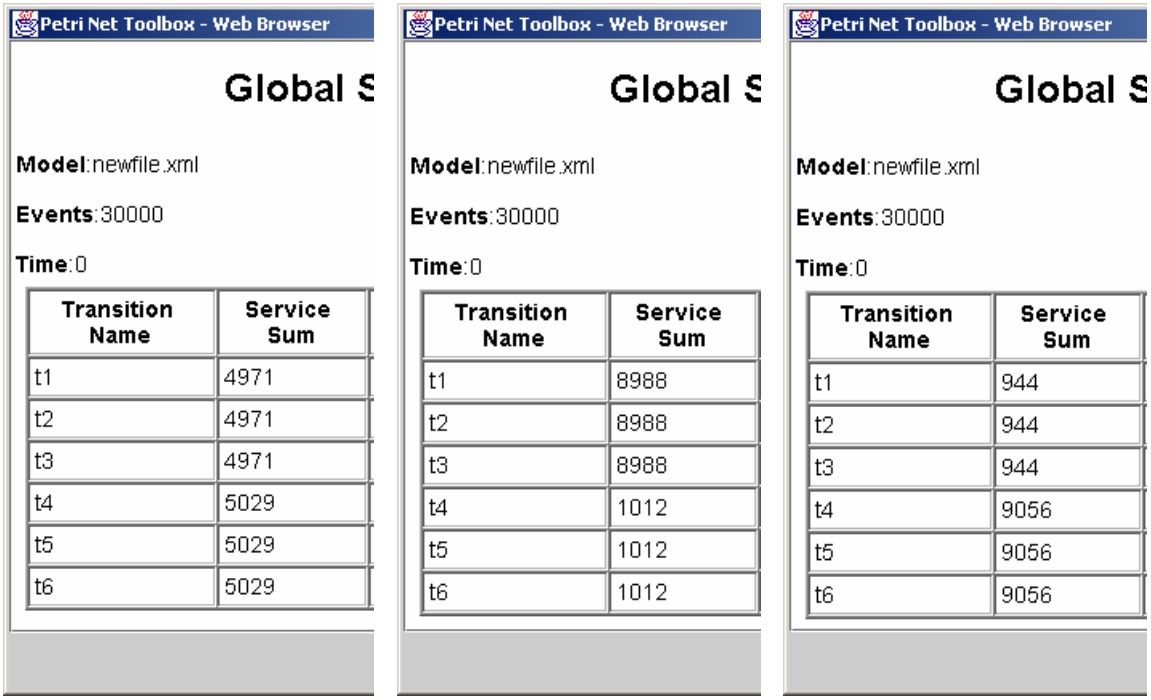


Fig. AP3.5.5. *Modelul de tip reţea Petri al sistemului de calcul prezentat în fig. AP3.5.2 utilizând arce inhibitoare.*



(a) (b) (c)

Fig. AP3.5.6. *Indicatorul **Service Sum** obţinut după simularea servirii a 10000 de taskuri:*
(a) *taskurile din ST_1 şi cele din ST_2 au aceeaşi probabilitate de servire;*
(b) *probabilitatea de servire a taskurilor din ST_1 este 0,9;*
(c) *probabilitatea de servire a taskurilor din ST_1 este 0,1.*

5. Pentru oricare dintre cele trei modele prezentate la punctul 4, putem verifica prin simulare în mediul ***Petri Net Toolbox*** funcţionarea corectă a sistemului de calcul. Deoarece servirii

complete a unui task de către fiecare dintre cele două procesoare îi corespunde apariţia a 3 evenimente (executarea a 3 tranziţii ale reţelei Petri), simularea servirii a 10.000 de taskuri este echivalentă cu apariţia a 30.000 de evenimente. Probabilităţile de servire a taskurilor din ST_1 şi ST_2 se asignează tranziţiilor t_1 şi t_4 ale reţelei Petri.

Numărul de taskuri servite complet de procesorul P_1 este redat de indicatorul ***Service Sum*** (numărul de executări) pentru tranziţia t_3 care este asociată terminării ultimei operaţii din ST_1. Pentru taskurile servite complet de procesorul P_2 consultăm acelaşi indicator, dar pentru tranziţia t_6 (fig. AP3.5.6). Se observă că raportul dintre numărul de taskuri din ST_1 şi numărul de taskuri din ST_2 servite complet este aproximativ egal cu raportul dintre probabilităţile de servire asignate tranziţiilor t_1 şi, respectiv, t_4.

AP3.6.

Se consideră un sistem de fabricaţie, adaptat după (Lewis et al., 1995), compus din două maşini: M_1 şi M_2, ambele fiind cu încărcare automată. Descărcarea celor două maşini este realizată de un robot R. Dacă R este liber şi există o piesă finalizată pe una din maşini, robotul efectuează transportul, descărcând maşina în cauză. Între M_1 şi M_2 există un depozit D în care pot fi stocate două piese. Pentru a putea fi prelucrate pe cele două maşini, piesele brute sunt fixate pe palete. În urma prelucrării, produsul finit este desfăcut de pe paletă, iar paleta liberă este returnată, pentru fixarea unei noi piese brute. În total se folosesc x palete. Se presupune că există un număr suficient de mare de piese brute care urmează să fie prelucrate. Întrucât piesele brute variază în dimensiuni, nu se poate face nici o precizare privind durata prelucrării pe maşina M_1, respectiv M_2. În fig. AP3.6.1. se prezintă schematizat structura procesului de fabricaţie. La momentul iniţial maşinile şi robotul sunt libere.

1. Să se construiască un model tip reţea Petri netemporizată pentru funcţionarea procesului de fabricaţie, specificând semnificaţia fizică a poziţiilor şi a tranziţiilor.
2. Să se investigheze posibilitatea producerii fenomenului de deadlock în funcţie de numărul paletelor folosite $x \in \{1, 2, 3, 4, 5\}$. În cazul apariţiei deadlock-ului, se vor detalia următoarele aspecte:
   (i) Se va preciza marcajul reţelei Petri pentru care se ajunge în deadlock precum şi secvenţele de executări de tranziţii care conduc la acesta.
   (ii) Se va preciza succesiunea de operaţii executate în sistemul de fabricaţie care conduce la deadlock şi se va explica modul de blocare circulară a resurselor.
3. Generalizând rezultatele de la punctul 2, să se investigheze posibilitatea producerii deadlock-ului în cazul unui număr arbitrar de palete. Să se determine numărul maxim de palete care asigură funcţionarea fără deadlock a sistemului, indiferent de succesiunea de operaţii executate.
4. Să se arate că indiferent de numărul de palete utilizate există întotdeauna o succesiune de operaţii care permite funcţionarea fără deadlock a sistemului.
5. În cazul funcţionării sistemului cu 4 palete să se precizeze ce modificări trebuie aduse modelului construit la punctul 1 astfel încât să se poată evita producerea fenomenului de deadlock.

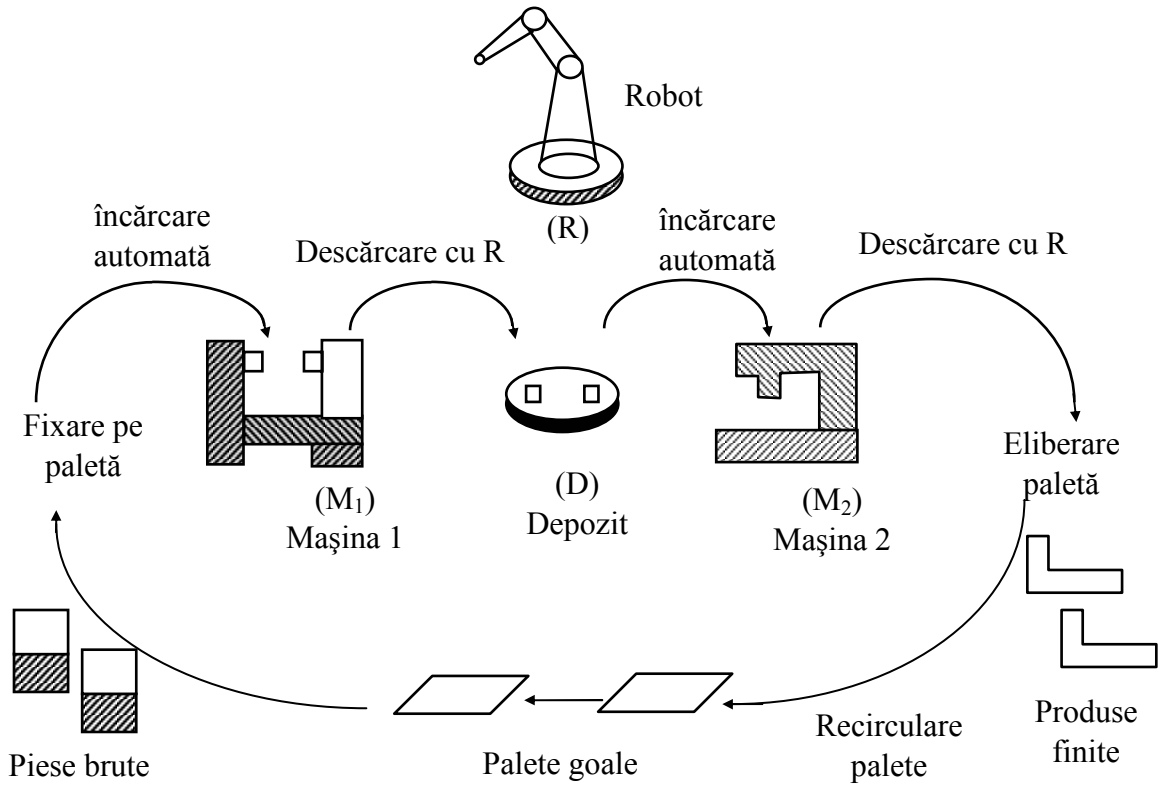


Fig. AP3.6.1. *Reprezentarea schematizată a fluxului tehnologic din AP3.6.*

Soluţie

1. Modelul de tip reţea Petri pentru sistemul de fabricaţie din fig. AP3.6.1. este prezentat în fig. AP3.6.2. Poziţiile sunt etichetate în conformitate cu semnificaţia lor fizică. Marcajul iniţial corespunde situaţiei în care se utilizează *x* palete pentru transportarea pieselor în sistem şi toate resursele sunt disponibile.

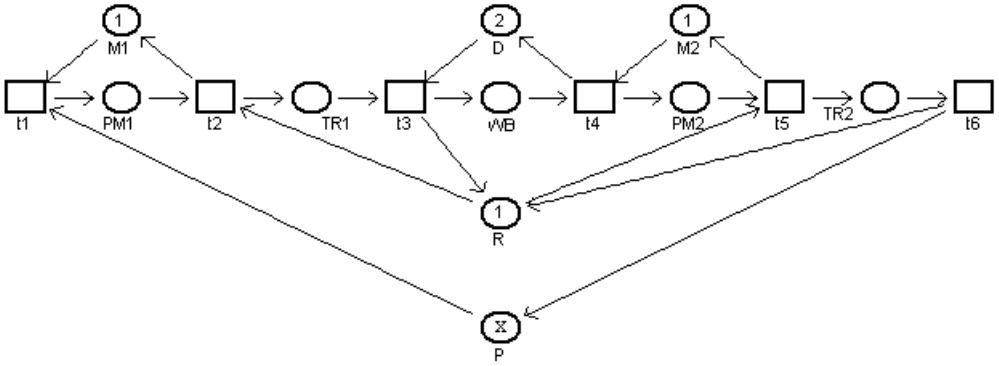


Fig. AP3.6.2. *Modelul tip reţea Petri netemporizată pentru sistemul de fabricaţie prezentat în fig. AP3.6.1.*

Semnificaţia fizică a poziţiilor şi tranziţiilor este următoarea:

- pentru poziţii: PM1 – prelucrarea unei piese pe maşina M_1; TR1 – descărcarea maşinii M_1 de către robot; WB – operaţia de stocarea în depozit; PM2 – prelucrarea unei piese

pe M_2; TR2 – descărcarea maşinii M_2 de către robot; P – resursă generală care modelează disponibilitatea paletelor;

- pentru tranziţii: t_1 – fixarea unei piese pe paletă şi începerea prelucrării pe M_1; t_2 – terminarea prelucrării pe M_1 şi începerea transportului în depozit; t_3 – terminarea transportului piesei de către robot şi începerea stocării piesei în depozit; t_4 – încărcarea unei piese din depozit pe maşina M_2 şi începerea prelucrării; t_5 – terminarea prelucrării pe M_2 şi începerea transportului de către robot la ieşire; t_6 – terminarea transportului la ieşire de către robot şi eliberare palete.

2. În cazul în care se utilizează 1, 2 sau 3 palete, fenomenul de deadlock nu apare, fapt vizibil în fig. AP3.6.3, în care este prezentat arborele de accesibilitate (construit cu ajutorul ***Petri Net Toolbox***) în cazul utilizării unui număr de 3 palete.

Petri Net Toolbox - Web Browser

PNToolbox - Coverability tree

From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To
M0	t1	M1	M1	t2	M2	M2	t1	M3	M2	t3	M4	M3	t3	M5	M4	t1	M5
M4	t4	M6	M5	t2	M7	M5	t4	M8	M6	t1	M8	M6	t5	M9	M7	t1	M10
M7	t3	M11	M7	t4	M12	M8	t2	M12	M8	t5	M13	M9	t1	M13	M9	t6	M0
M10	t3	M14	M10	t4	M15	M11	t1	M14	M11	t4	M16	M12	t1	M15	M12	t3	M16
M13	t6	M1	M14	t2	M17	M14	t4	M18	M15	t3	M18	M16	t1	M18	M16	t5	M19
M17	t4	M20	M18	t2	M20	M18	t5	M21	M19	t1	M21	M19	t4	M22	M19	t6	M4
M20	t3	M23	M21	t4	M24	M21	t6	M5	M22	t1	M24	M22	t6	M6	M23	t5	M25
M24	t6	M8	M25	t4	M26	M25	t6	M11	M26	t6	M16						

M[PM1,TR1,DO,PM2,TR2,M1,D,M2,R,P]

M0 = [0,0,0,0,0,1,2,1,1,3]	M1 = [1,0,0,0,0,0,2,1,1,2]	M2 = [0,1,0,0,0,1,2,1,0,2]
M3 = [1,1,0,0,0,0,0,2,1,0,1]	M4 = [0,0,1,0,0,1,1,1,1,2]	M5 = [1,0,1,0,0,0,1,1,1,1]
M6 = [0,0,0,1,0,1,2,0,1,2]	M7 = [0,1,1,0,0,1,1,1,0,1]	M8 = [1,0,0,1,0,0,2,0,1,1]
M9 = [0,0,0,0,1,1,2,1,0,2]	M10 = [1,1,1,0,0,0,1,1,0,0]	M11 = [0,0,2,0,0,1,0,1,1,1]
M12 = [0,1,0,1,0,1,2,0,0,1]	M13 = [1,0,0,0,1,0,2,1,0,1]	M14 = [1,0,2,0,0,0,0,1,1,0]
M15 = [1,1,0,1,0,0,2,0,0,0]	M16 = [0,0,1,1,0,1,1,0,1,1]	M17 = [0,1,2,0,0,1,0,1,0,0]
M18 = [1,0,1,1,0,0,1,0,1,0]	M19 = [0,0,1,0,1,1,1,1,0,1]	M20 = [0,1,1,1,0,1,1,0,0,0]
M21 = [1,0,1,0,1,0,1,1,0,0]	M22 = [0,0,0,1,1,1,2,0,0,1]	M23 = [0,0,2,1,0,1,0,0,1,0]
M24 = [1,0,0,1,1,0,2,0,0,0]	M25 = [0,0,2,0,1,1,0,1,0,0]	M26 = [0,0,1,1,1,1,1,0,0,0]

Ok Save

Fig. AP3.6.3. *Arborele de accesibilitate pentru sistemul de fabricaţie din fig. AP3.6.2 în cazul utilizării unui număr de 3 palete.*

Dacă se utilizează cel puţin 4 palete, apare fenomenul de blocare circulară a resurselor. Arborele de accesibilitate corespunzător utilizării a 4 palete (construit cu ajutorul ***Petri Net Toolbox***) este prezentat în fig. AP3.6.4, în care marcajul M_{29} = (0, 1, 2, 1, 0, 1, 0, 0, 0, 0) este

marcajul de deadlock. Urmărind secvenţele de executări de tranziţii din arborele de accesibilitate se observă că sistemul ajunge în deadlock numai din starea M_{27} (în care ambele spaţii din depozit sunt ocupate, o piesă este în curs de prelucrare pe M_1 şi alta pe M_2) când sunt validate tranziţiile t_2 şi t_5. Executarea tranziţiei t_5 înseamnă eliberarea maşinii M_2 şi posibilitatea continuării fluxului de producţie, în timp ce executarea lui t_2 revine la ocuparea robotului cu piesa descărcată de pe M_1 şi, respectiv, la invalidarea operaţiei de eliberare a maşinii M_2. Blocajul circular de resurse constă în faptul că M_2 aşteaptă R pentru descărcare, D aşteaptă loc liber pe M_2 şi R aşteaptă loc liber în D.

Petri Net Toolbox - Web Browser

PNToolbox - Coverability tree

From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To
M0	t1	M1	M1	t2	M2	M2	t1	M3	M2	t3	M4	M3	t3	M5	M4	t1	M5
M4	t4	M6	M5	t2	M7	M5	t4	M8	M6	t1	M8	M6	t5	M9	M7	t1	M10
M7	t3	M11	M7	t4	M12	M8	t2	M12	M8	t5	M13	M9	t1	M13	M9	t6	M0
M10	t3	M14	M10	t4	M15	M11	t1	M14	M11	t4	M16	M12	t1	M15	M12	t3	M16
M13	t6	M1	M14	t2	M17	M14	t4	M18	M15	t3	M18	M16	t1	M18	M16	t5	M19
M17	t1	M20	M17	t4	M21	M18	t2	M21	M18	t5	M22	M19	t1	M22	M19	t4	M23
M19	t6	M4	M20	t4	M24	M21	t1	M24	M21	t3	M25	M22	t4	M26	M22	t6	M5
M23	t1	M26	M23	t6	M6	M24	t3	M27	M25	t1	M27	M25	t5	M28	M26	t6	M8
M27	t2	M29	M27	t5	M30	M28	t1	M30	M28	t4	M31	M28	t6	M11	M30	t4	M32
M30	t6	M14	M31	t1	M32	M31	t6	M16	M32	t6	M18						

M[PM1,TR1,DO,PM2,TR2,M1,D,M2,R,P]

M0 = [0,0,0,0,0,1,2,1,1,4]	M1 = [1,0,0,0,0,0,2,1,1,3]	M2 = [0,1,0,0,0,1,2,1,0,3]
M3 = [1,1,0,0,0,0,2,1,0,2]	M4 = [0,0,1,0,0,1,1,1,1,3]	M5 = [1,0,1,0,0,0,1,1,1,2]
M6 = [0,0,0,1,0,1,2,0,1,3]	M7 = [0,1,1,0,0,1,1,1,0,2]	M8 = [1,0,0,1,0,0,2,0,1,2]
M9 = [0,0,0,0,1,1,2,1,0,3]	M10 = [1,1,1,0,0,0,1,1,0,1]	M11 = [0,0,2,0,0,1,0,1,1,2]
M12 = [0,1,0,1,0,1,2,0,0,2]	M13 = [1,0,0,0,1,0,2,1,0,2]	M14 = [1,0,2,0,0,0,0,1,1,1]
M15 = [1,1,0,1,0,0,2,0,0,1]	M16 = [0,0,1,1,0,1,1,0,1,2]	M17 = [0,1,2,0,0,1,0,1,0,1]
M18 = [1,0,1,1,0,0,1,0,1,1]	M19 = [0,0,1,0,1,1,1,1,0,2]	M20 = [1,1,2,0,0,0,0,1,0,0]
M21 = [0,1,1,1,0,1,1,0,0,1]	M22 = [1,0,1,0,1,0,1,1,0,1]	M23 = [0,0,0,1,1,1,2,0,0,2]
M24 = [1,1,1,1,0,0,1,0,0,0]	M25 = [0,0,2,1,0,1,0,0,1,1]	M26 = [1,0,0,1,1,0,2,0,0,1]
M27 = [1,0,2,1,0,0,0,0,1,0]	M28 = [0,0,2,0,1,1,0,1,0,1]	M29 = [0,1,2,1,0,1,0,0,0,0]
M30 = [1,0,2,0,1,0,0,1,0,0]	M31 = [0,0,1,1,1,1,1,0,0,1]	M32 = [1,0,1,1,1,0,1,0,0,0]

Ok Save

Fig. AP3.6.4. *Arborele de accesibilitate pentru sistemul de fabricaţie din fig. AP3.6.2 în cazul utilizării unui număr de 4 palete.*

3. Generalizând rezultatele obţinute la punctul anterior, se observă că în situaţia în care se utilizează 4 sau mai multe palete, în sistemul de fabricaţie apare blocarea circulară a resurselor. Numărul maxim de palete pentru care acest fenomen nu se produce este 3.

4. Din discuţia anterioară rezultă că fenomenul de deadlock nu s-ar produce (indiferent de numărul de palete din sistem) dacă, ori de câte ori sunt validate simultan tranziţiile t_2 şi t_5 (adică robotul este în situaţia de a descărca oricare dintre maşini), s-ar executa t_5 (robotul eliberează M_2).

5. Constatarea de la punctul anterior este tot una cu existenţa unui mecanism de asignare a priorităţilor ce asigură prioritate în executarea lui t_5 în raport cu executarea lui t_2, atunci când ambele sunt validate. Pentru situaţia utilizării unui număr de 4 palete, arborele de accesibilitate (construit cu ajutorul ***Petri Net Toolbox***) este prezentat în fig. AP3.6.5; se poate observa că nu mai există marcaj de dealock.

Petri Net Toolbox - Web Browser

PNToolbox - Coverability tree

From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To
M0	t1	M1	**M1**	**t2**	**M2**	M2	t1	M3	**M2**	**t3**	**M4**	M3	t3	M5	**M4**	**t1**	**M5**
M4	t4	M6	**M5**	**t2**	**M7**	M5	t4	M8	**M6**	**t1**	**M8**	M6	t5	M9	**M7**	**t1**	**M10**
M7	t3	M11	**M7**	**t4**	**M12**	M8	t5	M13	**M9**	**t1**	**M13**	M9	t6	M0	**M10**	**t3**	**M14**
M10	t4	M15	**M11**	**t1**	**M14**	M11	t4	M16	**M12**	**t1**	**M15**	M12	t3	M16	**M13**	**t6**	**M1**
M14	t2	M17	**M14**	**t4**	**M18**	M15	t3	M18	**M16**	**t1**	**M18**	M16	t5	M19	**M17**	**t1**	**M20**
M17	t4	M21	**M18**	**t5**	**M22**	M19	t1	M22	**M19**	**t4**	**M23**	M19	t6	M4	**M20**	**t4**	**M24**
M21	t1	M24	**M21**	**t3**	**M25**	M22	t4	M26	**M22**	**t6**	**M5**	M23	t1	M26	**M23**	**t6**	**M6**
M24	t3	M27	**M25**	**t1**	**M27**	M25	t5	M28	**M26**	**t6**	**M8**	M27	t5	M29	**M28**	**t1**	**M29**
M28	t4	M30	**M28**	**t6**	**M11**	M29	t4	M31	**M29**	**t6**	**M14**	M30	t1	M31	**M30**	**t6**	**M16**
M31	t6	M18															

M[PM1,TR1,DO,PM2,TR2,M1,D,M2,R,P]

M0 = [0,0,0,0,0,1,2,1,1,4]	M1 = [1,0,0,0,0,0,2,1,1,3]	M2 = [0,1,0,0,0,1,2,1,0,3]
M3 = [1,1,0,0,0,0,2,1,0,2]	M4 = [0,0,1,0,0,1,1,1,1,3]	M5 = [1,0,1,0,0,0,1,1,1,2]
M6 = [0,0,0,1,0,1,2,0,1,3]	M7 = [0,1,1,0,0,1,1,1,0,2]	M8 = [1,0,0,1,0,0,2,0,1,2]
M9 = [0,0,0,0,1,1,2,1,0,3]	M10 = [1,1,1,0,0,0,1,1,0,1]	M11 = [0,0,2,0,0,1,0,1,1,2]
M12 = [0,1,0,1,0,1,2,0,0,2]	M13 = [1,0,0,0,1,0,2,1,0,2]	M14 = [1,0,2,0,0,0,0,1,1,1]
M15 = [1,1,0,1,0,0,2,0,0,1]	M16 = [0,0,1,1,0,1,1,0,1,2]	M17 = [0,1,2,0,0,1,0,1,0,1]
M18 = [1,0,1,1,0,0,1,0,1,1]	M19 = [0,0,1,0,1,1,1,1,0,2]	M20 = [1,1,2,0,0,0,0,1,0,0]
M21 = [0,1,1,1,0,1,1,0,0,1]	M22 = [1,0,1,0,1,0,1,1,0,1]	M23 = [0,0,0,1,1,1,2,0,0,2]
M24 = [1,1,1,1,0,0,1,0,0,0]	M25 = [0,0,2,1,0,1,0,0,1,1]	M26 = [1,0,0,1,1,0,2,0,0,1]
M27 = [1,0,2,1,0,0,0,0,1,0]	M28 = [0,0,2,0,1,1,0,1,0,1]	M29 = [1,0,2,0,1,0,0,1,0,0]
M30 = [0,0,1,1,1,1,1,0,0,1]	M31 = [1,0,1,1,1,0,1,0,0,0]	

Ok Save

Fig. AP3.6.5. *Arborele de accesibilitate pentru sistemul de fabricaţie din fig. AP3.6.2. în cazul utilizării unui număr de 4 palete şi setarea de priorităţi între t_2 şi t_5.*

Capitolul 4

Controlul procedural al sistemelor cu evenimente discrete

Breviar teoretic

BT4.1. Specificații de proiectare pentru structurile de conducere

Un proces tehnic cu funcționarea pilotată de evenimente este constituit dintr-o *mulțime de resurse* care sunt utilizate pentru a efectua o *succesiune de operații*. Realizarea fiecărei operații necesită *alocarea* uneia sau mai multor resurse care sunt *eliberate* după încheierea operației respective.

Procesul are drept intrare un număr de *clienți* (nume generic desemnând entități a căror natură fizică depinde de specificul procesului). Fiecare operație realizată de proces reprezintă un anumit tip de *serviciu* prestat unui client. În urma parcurgerii *secvențiale* a întregii succesiuni de operații, un client este *servit complet* şi poate părăsi procesul. Astfel, *ieşirea* procesului constă în numărul de clienți serviți complet.

În cazul general, diferiți clienți pot necesita diferite succesiuni de operații. În aplicațiile practice uzuale, numărul de succesiuni diferite de operații (linii de servire diferite) este de ordinul unităților.

Problematica conducerii unui astfel de proces constă în a asigura îndeplinirea următoarelor condiții de funcționare:

- **(C1)** corectitudinea succesiunii de operații pentru toți clienții;
- **(C2)** corectitudinea alocării şi eliberării resurselor necesare de fiecare operație în parte;
- **(C3)** prestarea unui anumit tip de serviciu de îndată ce resursele necesare pentru operația respectivă sunt disponibile (adică, maximizarea numărului de clienți aflați în curs de servire, în etape diferite);
- **(C4)** repetabilitatea prestării serviciilor fără blocaje circulare datorate utilizării partajate a unora dintre resurse.

O *structură de conducere* (proces + controler), reprezentată prin schema bloc din fig. BT4.1.1, ce asigură satisfacerea condițiilor de funcționare **(C1) – (C4)** exploatează numai *proprietăți de tip logic* (independente de durata operațiilor) şi, drept consecință, descrierea acestei structuri poate fi abordată prin formalismul rețelelor Petri netemporizate.

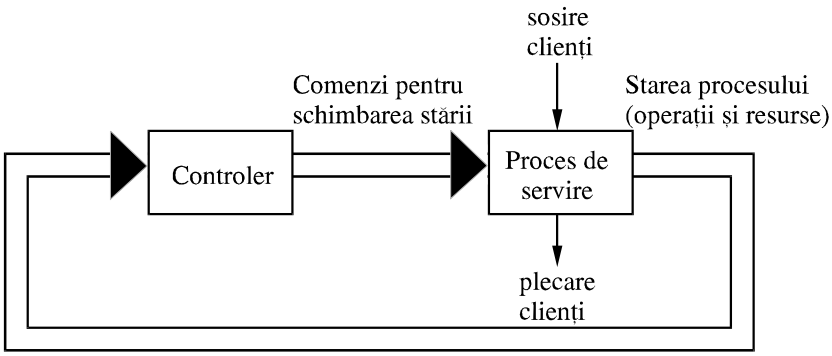


Fig. BT4.1.1. *Reprezentarea sub formă de schemă bloc a structurii de conducere a unui proces cu evenimente discrete.*

BT4.2. Proiectarea structurilor de conducere prin tehnici de sinteză hibridă

Sinteza hibridă furnizează o rețea Petri care modelează atât efectuarea operațiilor cât şi utilizarea resurselor. Sinteza *hibridă* combină avantajele unei sinteze *descendente* (eng. *top-down*), urmată de o sinteză *ascendentă* (eng. *bottom-up*). În general, sinteza *descendentă* se aplică în rafinarea necesară modelării tuturor detaliilor privind modul în care se succed operațiile, iar sinteza *ascendentă* permite modelarea modului de utilizare a resurselor.
În acest scop resursele se clasifică în:

- *Resurse generale* – care sunt alocate unui client în momentul intrării acestuia în sistem, se utilizează pe tot parcursul servirii şi sunt eliberate în momentul când servirea este completă (şi clientul părăseşte sistemul). O resursă generală poate fi *simplă* sau *multiplă*. În acest din urmă caz, resursa dispune de mai multe unități fizice similare, care pot fi alocate mai multor clienți de acelaşi tip.
- *Resurse specifice* – care sunt alocate unui client numai pentru prestarea anumitor tipuri de servicii (operații) şi sunt eliberate de îndată ce acele servicii au fost efectuate. O resursă specifică este simplă. Ea poate fi utilizată *nepartajat*, pentru realizarea unei singure operații (sau secvențe de operații), sau *partajat*, pentru realizarea mai multor operații (sau secvențe de operații).
- *Resurse de stocare* (tip depozit sau tampon) care pun la dispoziție un spațiu fizic (a cărui natură depinde de tipul procesului) în care un număr limitat de clienți pot aştepta servirea de către o resursă specifică. O resursă de stocare este în general multiplă, întrucât oferă mai multe zone disponibile de aşteptare în care se pot găsi mai mulți clienți.

BT4.2.1. Rafinarea operațiilor prin sinteză descendentă

BT4.2.1.1. Prezentarea generală a procedurii de rafinare

Modelul global considerat *inițial* (de la care porneşte expandarea) este o rețea Petri cu o topologie foarte simplă, *mărginită* (*sigură*), *viabilă* şi *reversibilă* ce conține informații referitoare la:

- *operații* (detaliate sumar);

- *resurse generale* (care, fiind utilizate pe tot parcursul servirii, nu sunt afectate de detalierea sumară a operațiilor).

În acest stadiu al construirii modelului, pozițiile asociate operațiilor nu sunt marcate, iar pozițiile asociate resurselor generale sunt marcate cu $m \geq 1$ jetoane. Pentru un anumit tip de resursă generală, marcajul inițial m precizează multiplicitatea resursei generale respective (adică, numărul total de unități fizice ale resursei generale respective care sunt disponibile). Astfel, numărul de clienți aflați în curs de servire, care necesită un acelaşi tip de resursă generală, nu poate depăşi valoarea m. De aceea, în situația când procesul nu posedă resurse generale ca şi *entități fizice*, în modelare se utilizează totuşi o poziție de acest tip (chiar dacă nu are un corespondent fizic concret) pentru a putea limita, prin marcajul inițial m, numărul clienților acceptați în sistem pentru diferite etape ale servirii.

În cazurile când nu se poate preciza exact valoarea lui $m \geq 1$, această valoare va fi fixată ulterior, în *sinteza ascendentă*, conform unor criterii ce vor fi prezentate în paragraful următor.

În fig. BT4.2.1.(a) este reprezentat modelul cu o detaliere sumară a operațiilor necesare servirii unui anumit tip de clienți, de-a lungul unei singure linii ce utilizează resursa generală R_g. Facem precizarea că rețeaua Petri ce constituie punctul de plecare al sintezei descendente va conține k modele de factura celui din fig. BT4.2.1.(a) atunci când sistemul conține k linii (fluxuri) de servire, conform reprezentării grafice din fig. BT4.2.1.(b). Clienții serviți de aceste k linii pot fi de tipuri diferite, sau de acelaşi tip.

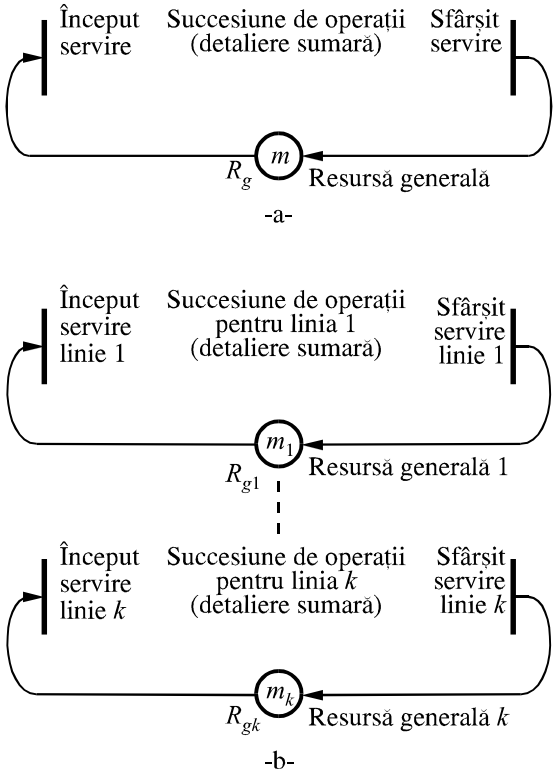


Fig. BT4.2.1. *Modelul de la care se începe sinteza ascendentă prin rafinarea operațiilor*
(a) *o singură linie (flux) de servire;* (b) *k linii (fluxuri) de servire.*

BT4.2.1.2. Module standard utilizate în rafinare

Procedeul de rafinare a operațiilor prin sinteză descendentă se bazează pe expandarea unor tranziții sau poziții. În urma unei expandări, o tranziție (poziție) a rețelei inițiale este înlocuită de o subrețea ce posedă un anumit tip de comportare, utilizând aşa-numite *blocuri T* (respectiv *P*) *corect formate*.

Un *bloc T* (tip tranziție) este definit ca fiind o rețea Petri, notată $\boldsymbol{N}_T$, a cărei topologie conține:

- o *tranziție inițială* (de tip sursă), notată t_{in};
- o *tranziție finală* (de tip receptor), notată t_{fi}.

Pornind de la $\boldsymbol{N}_T$, se construieşte *rețeaua extinsă* asociată blocului *T* (sau, echivalent, asociată rețelei $\boldsymbol{N}_T$) notată $\boldsymbol{N}_{T_e}$, prin adăugarea unei noi poziții, notată p_e, în maniera următoare:

- singura tranziție de ieşire a lui p_e este t_{in}, adică $\{p_e\}^\bullet = t_{in}$;
- singura tranziție de intrare a lui p_e este t_{fi}, adică ${}^\bullet\{p_e\} = t_{fi}$;
- inițial, poziția p_e poate conține cel mult *k* jetoane, adică $M_0(p_e) \le k$.

Să notăm prin $L(\boldsymbol{N}_{T_e}, M_0)$ mulțimea tuturor secvențelor de executări din rețeaua $\boldsymbol{N}_{T_e}$, pornind de la marcajul inițial M_0. Dată fiind o secvență de executări σ, să notăm prin $\overline{\sigma}(t)$ numărul de executări a tranziției *t* din această secvență.

Se spune că blocul *T* (sau rețeaua $\boldsymbol{N}_T$) este un *bloc T de ordin k, corect format în raport cu tranzițiile* t_{in} şi t_{fi}, dacă rețeaua extinsă $\boldsymbol{N}_{T_e}$ posedă următoarele proprietăți:

(i) Tranziția t_{in} este viabilă.

(ii) Pentru orice secvență $\sigma_1 \in L(\boldsymbol{N}_{T_e}, M_0)$ în care avem inegalitatea:

$$\overline{\sigma}_1(t_{in}) > \overline{\sigma}_1(t_{fi}),$$

există o secvență σ_2 care nu conține nici o executare a lui t_{in}, astfel încât secvența compusă $(\sigma_1\sigma_2)$ să aparțină mulțimii de secvențe $L(\boldsymbol{N}_{T_e}, M_0)$ şi să avem egalitatea:

$$\overline{(\sigma_1\sigma_2)}(t_{in}) = \overline{(\sigma_1\sigma_2)}(t_{fi}).$$

(iii) Pentru orice secvență $\sigma \in L(\boldsymbol{N}_{T_e}, M_0)$, avem inegalitatea:

$$\overline{\sigma}(t_{in}) \ge \overline{\sigma}(t_{fi})$$

Cu alte cuvinte, condițiile (*i*) – (*iii*) precizează următorul tip de comportare al rețelei $\boldsymbol{N}_{T_e}$: tranziția t_{in} nu se va bloca niciodată (condiția (*i*)) şi poate devansa (ca număr de executări) pe t_{fi}, iar pentru tranziția t_{fi} va exista întotdeauna o posibilitate de a recupera devansul (ca număr de executări) în raport cu t_{in}.

Modul de operare al rețelei extinse $\boldsymbol{N}_{T_e}$ asociată unui bloc *T* corect format, face ca un astfel de bloc să poată fi substituit unei tranziții. În acest scop definim noțiunea de *tranziție validabilă de ordin k*. Se spune că o tranziție *t* dintr-o rețea Petri $(\boldsymbol{N}, M_0)$ este validabilă de ordin *k*, dacă există un marcaj *M*, accesibil din M_0, astfel încât toate pozițiile predecesor ale lui *t* conțin cel puțin *k* jetoane, adică $\forall p \in {}^\bullet\{t\}: M(p) \ge k$.

În exploatarea tehnicilor de sinteză prin rafinare ce utilizează blocuri *T* corect formate, o importanță deosebită este deținută de modul de conservare a unor proprietăți comportamentale în urma expandării, după cum reiese din următoarea teoremă.

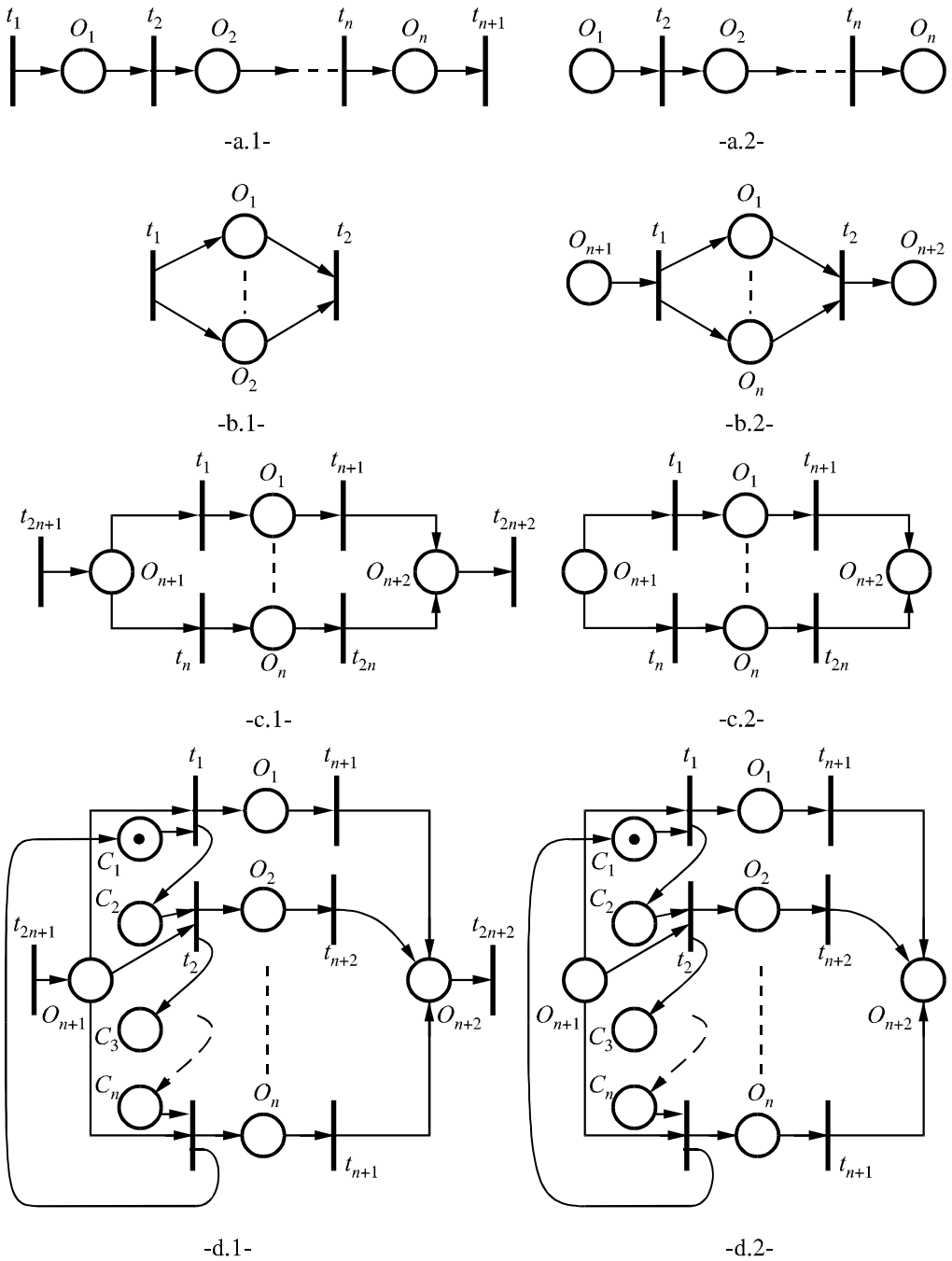


Fig. BT4.2.2. *Definiția modulelor standard utilizate în rafinare*
(a) *Operații secvențiale* (a.1 – *bloc T,* a.2 – *bloc P*);
(b) *Operații paralele* (b.1 – *bloc T,* b.2 – bloc *P*);
(c) *Operații la alegere, conflictuale* (c.1 – *bloc T,* c.2 – *bloc P*);
(d) *Operații la alegere, neconflictuale* (d.1 – *bloc T,* d.2 – *bloc P*).

Teorema 4.2.1. Fie o reţea Petri ($\boldsymbol{N}$, M_0) şi fie o tranziţie t_0 a acestei reţele, care este validabilă de ordin k şi nu este validabilă de ordin k+1 pentru marcajul iniţial M_0. Fie $\boldsymbol{N}'$ reţeaua ce rezultă prin substituirea lui t_0 cu reţeaua $\boldsymbol{N}_T$ ce defineşte un bloc T de ordin k, corect format.

a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci $\boldsymbol{N}'$ este mărginită (sigură).

b) Dacă $\boldsymbol{N}$ este viabilă, atunci $\boldsymbol{N}'$ este viabilă.

Prin analogie, se definesc *blocuri P* (tip poziţie) corect formate care pot substitui o poziţie, păstrând proprietăţile de mărginire (siguranţă) şi viabilitate (similar rezultatului enunţat în **Teorema 4.2.1**).

Adăugarea tuturor *detaliilor* privind operaţiile se realizează expandând poziţiile sau tranziţiile cu ajutorul următoarelor *module standard,* reprezentate sub formă de *blocuri T* sau *blocuri P* corect formate, după cum urmează:

a) *Modul de operaţii secvenţiale* – reprezentat în fig. BT4.2.2.(a) (a.1 – bloc *T*, a.2 – bloc *P*).

b) *Modul de operaţii paralele* – reprezentat în fig. BT4.2.2.(b) (b.1 – bloc *T*, b.2 – bloc *P*).

c) *Modul de operaţii la alegere, conflictuale*, reprezentat în fig. BT4.2.2.(c) (c.1 – bloc *T*, c.2 – bloc *P*).

d) *Modul de operaţii la alegere, neconflictuale*, reprezentat în fig. BT4.2.2.(d) (d.1 – bloc *T*, d.2 – bloc *P*). Acest modul diferă de precedentul prin faptul că situaţiile conflictuale dintre operaţiile la alegere sunt rezolvate de structura de control modelată prin poziţiile C_1,...,C_n şi arcele aferente.

Marcajul iniţial al tuturor modulelor standard este nul, exceptând o singură poziţie din structura de control ce elimină conflictele dintre operaţiile la alegere în modulele tip d) (adică, în fig. BT4.2.2.(d.1) şi (d.2), poziţia notată C_1).

Succesul sintezei descendente prin rafinare, bazată pe aceste module standard, este garantat de următoarea teoremă.

Teorema 4.2.2. Fie o reţea Petri $\boldsymbol{N}$. Fie $\boldsymbol{N}'$ reţeaua rezultată prin substituirea unei tranziţii (respectiv poziţii) a lui $\boldsymbol{N}$ cu unul din modulele standard din fig. BT4.2.2, reprezentat sub formă de bloc *T* (respectiv, bloc *P*).

a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci $\boldsymbol{N}'$ este mărginită (sigură).

b) Dacă $\boldsymbol{N}$ este viabilă, atunci $\boldsymbol{N}'$ este viabilă.

c) Dacă $\boldsymbol{N}$ este reversibilă, atunci $\boldsymbol{N}'$ este reversibilă.

Prin expandarea repetată, se obţine gradul de detaliere dorit pentru modelarea operaţiilor. Fiecărei operaţii îi corespunde o poziţie pe aşa-numitele *drumuri* sau *căi de operaţii* a liniilor (fluxurilor) de servire. Aceste căi de operaţii corespund fluxurilor de servire a tuturor tipurilor de clienţi care vizitează sistemul. În plus, reţeaua ce rezultă în finalul sintezei descendente este mărginită (sigură), viabilă şi reversibilă, întrucât fiecare pas al rafinării păstrează aceste proprietăţi, conform **Teoremei 4.2.2**.

BT4.2.2. Ataşarea resurselor prin sinteză ascendentă

BT4.2.2.1. Prezentarea generală a procedurii de ataşare a resurselor

Ca principiu fundamental, sinteza ascendentă permite ataşarea resurselor specifice la modelul obținut în urma sintezei descendente (prezentată în paragraful anterior). Prin aplicarea acestui procedeu se păstrează structura construită prin sinteza descendentă (tranziții, poziții nemarcate pentru operații, poziții marcate pentru resurse generale) şi se adaugă *numai* poziții marcate (cu arcele aferente) pentru modelarea utilizării resurselor specifice. Rezultatele ce vor fi prezentate mai jos garantează *mărginirea* (*siguranța*), *viabilitatea* şi *reversibilitatea* rețelei Petri ce constituie modelul final al funcționării sistemului.

Pornind de la modelul tip rețea Petri netemporizată rezultat în urma detalierii operațiilor prin sinteză descendentă, ataşarea resurselor se realizează conform următorilor *paşi*:
Pas 1. Ataşarea resurselor specifice nepartajate
Pas 2. Ataşarea resurselor de stocare
  Subpas 2.1. Ataşarea resurselor de stocare nepartajate
  Subpas 2.2. Ataşarea resurselor de stocare partajate
Pas 3. Ataşarea resurselor specifice partajate
  Subpas 3.1. Ataşarea resurselor specifice partajate paralel
  Subpas 3.2. Ataşarea resurselor specifice partajate secvențial
  Subpas 3.3. Ataşarea resurselor specifice partajate paralel şi secvențial
În cele ce urmează vom schița fundamentele teoretice pe care se bazează fiecare din aceşti paşi ai sintezei ascendente.

BT4.2.2.2. Detalierea Pasului 1 (ataşarea resurselor specifice nepartajate)

Se consideră rețeaua $\boldsymbol{N}$ care include o poziție O, modelând o operație servită de o resursă specifică nepartajată R_n. Modelarea utilizării nepartajate a lui R_n se face printr-o poziție cu marcajul inițial $M_0(R_n) = 1$, conectată la rețeaua $\boldsymbol{N}$ conform fig. BT4.2.3.(a); R_n se alocă la începutul operației care o necesită şi se eliberează la sfârşitul acestei operații.

Într-o manieră cu totul similară, resursa specifică nepartajată poate servi o secvență de operații efectuate asupra aceluiaşi tip de clienți. Modul de ataşare a lui R_n în acest caz este ilustrat în fig. BT4.2.3.(b).

Teorema 4.2.3. Fie $\boldsymbol{N'}$ rețeaua rezultată din $\boldsymbol{N}$ prin adăugarea poziției R_n şi a arcelor de conexiune, cu $M_0(R_n) = 1$ (conform fig. BT4.2.3.(a) sau (b)).
a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci $\boldsymbol{N'}$ este mărginită (sigură).
b) Dacă $\boldsymbol{N}$ este viabilă, atunci $\boldsymbol{N'}$ este viabilă.
c) Dacă $\boldsymbol{N}$ este reversibilă, atunci $\boldsymbol{N'}$ este reversibilă.

Atragem atenția asupra generalității **Teoremei 4.2.3**, care rămâne valabilă şi în cazul când una sau mai multe operații din secvența $O_1 \ldots O_j$ au deja ataşate alte resurse specifice nepartajate. Această ataşare prealabilă a altor resurse specifice nepartajate este simbolizată în fig. BT4.2.3.(b) prin arcele trasate cu linie întreruptă, resursele nemaifiind figurate pentru a nu încărca reprezentarea grafică. Un model de această factură este util atunci când efectuarea unei operații necesită două sau mai multe resurse specifice nepartajate.

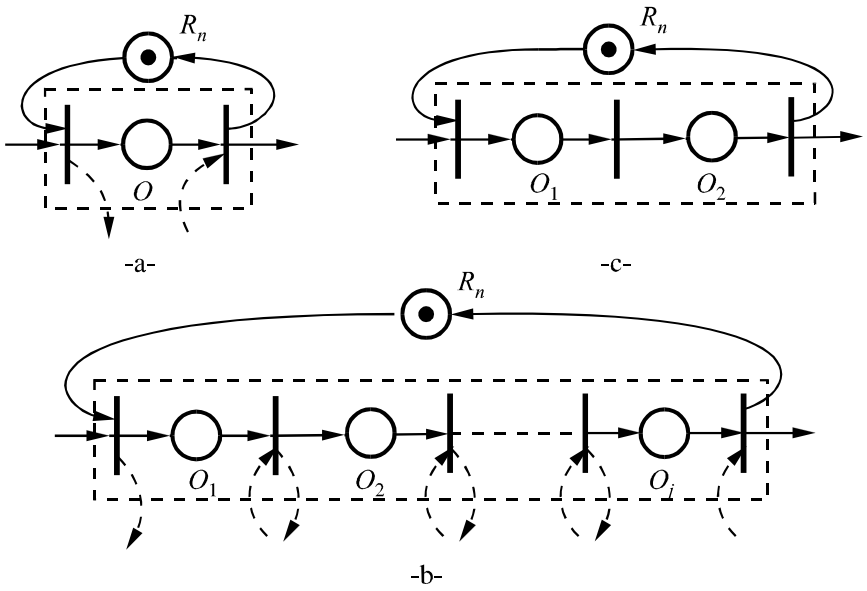


Fig. BT4.2.3. *Ilustrarea ataşării la reţeaua **N** a resursei specifice, nepartajate R_n*
(a) *R_n serveşte o singură operaţie O;* (b) *R_n serveşte o secvenţă de operaţii O_1, O_2,..., O_j;*
(c) *R_n serveşte operaţia O_2 şi este alocată cu anticipare.*

Facem precizarea că alocarea lui R_n se poate face şi *cu anticipare*, adică înainte ca R_n să fie *efectiv* utilizată. De exemplu, în fig. BT4.2.3.(c), R_n poate fi alocată la începerea operaţiei O_1, deşi ea este utilizată efectiv abia de către operaţia O_2. Alocarea cu anticipare a resurselor nepartajate este utilizată numai în anumite situaţii ce vor fi detaliate şi motivate în *Pasul* 3. **Teorema 4.2.3** îşi păstrează valabilitatea şi în cazul alocării cu anticipare a resurselor specifice nepartajate.

La încheierea *Pasului* 1 al sintezei ascendente, toate resursele specifice nepartajate sunt ataşate modelului, conservând proprietăţile de mărginire (siguranţă), viabilitate şi reversibilitate.

BT4.2.2.3. Detalierea Pasului 2 (ataşarea resurselor de stocare)

Subpasul 2.1. Ataşarea resurselor de stocare nepartajate

Se consideră reţeaua ***N*** care include o poziţie, O, modelând aşteptarea clienţilor de un anumit tip într-o resursă de stocare (depozit sau tampon) nepartajată S_n. Modelarea utilizării lui S_n se face printr-o poziţie cu marcajul iniţial $M_0(S_n) = q$, conectată la reţeaua ***N*** conform fig. BT4.2.4.(a). Valoarea $q \geq 2$ reprezintă capacitatea resursei de stocare, adică numărul maxim de clienţi ce pot aştepta (cazul $q = 1$ nu are relevanţă practică) .

Facem observaţia (de altfel, firească) că o poziţie asociată unei operaţii de aşteptare nu necesită nici o altă resursă în afara resursei de stocare (spre deosebire de poziţiile asociate unor operaţii de procesare, care pot necesita două sau mai multe resurse).

Ca şi în cazul resurselor specifice nepartajate, resursele de stocare nepartajate pot fi alocate cu anticipare.

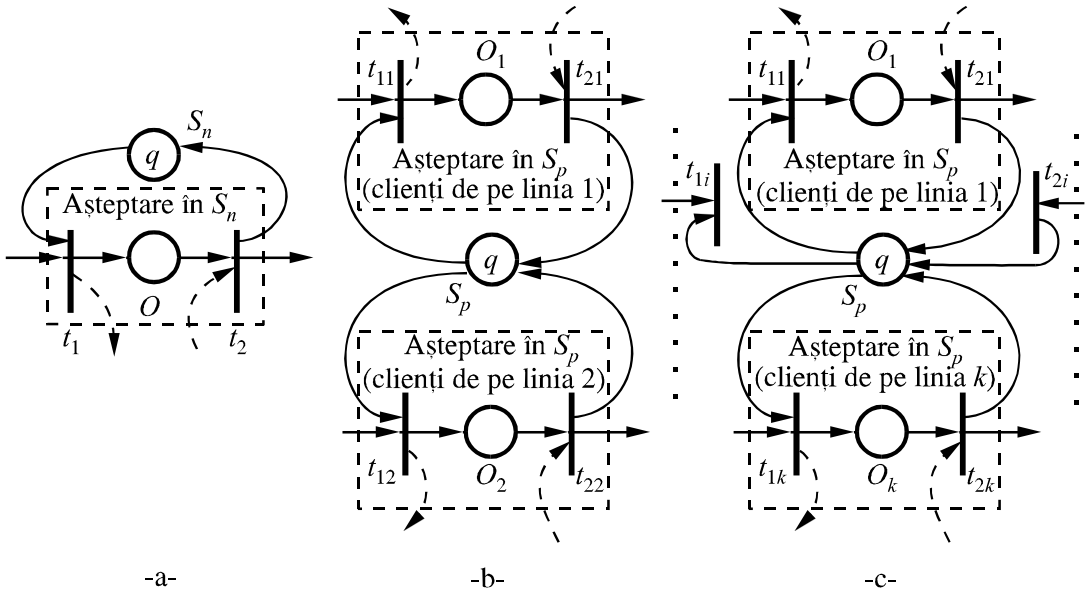


Fig. BT4.2.4. *Ilustrarea ataşării la rețeaua **N** a unei resurse de stocare*
(a) *Resursă de stocare nepartajată* (S_n);
(b) *Resursă de stocare partajată* (S_p) *între clienți de acelaşi tip, serviți pe două linii* (*fluxuri*) *diferite*;
(c) *Resursă de stocare partajată* (S_p) *între clienți de acelaşi tip, serviți pe k linii* (*fluxuri*) *diferite.*

Teorema 4.2.4. Fie ***N′*** rețeaua rezultată din ***N*** prin adăugarea poziției S_n şi a arcelor de conexiune, cu $M_0(S_n)= q \geq 2$ (conform fig. BT4.2.4.(a)).
a) Dacă ***N*** este mărginită , atunci ***N′*** este mărginită.
b) Dacă ***N*** este viabilă, atunci ***N′*** este viabilă.
c) Dacă ***N*** este reversibilă, atunci ***N′*** este reversibilă.

Prin aplicarea repetată a **Teoremei 4.2.4**, la încheierea *Subpasului 2.1* al sintezei ascendente, toate resursele de stocare nepartajate sunt ataşate modelului, conservând proprietățile de mărginire, viabilitate şi reversibilitate.

Subpasul 2.2. Ataşarea resurselor de stocare partajate

Se consideră rețeaua ***N*** care include două poziții, O_1 şi O_2, modelând aşteptarea, într-o resursă de stocare partajată S_p, a unor clienți de acelaşi tip, serviți pe două linii (fluxuri) diferite. Modelarea utilizării lui S_p se face printr-o poziție cu marcajul inițial $M_0(S_p)= q \geq 2$ conectată la rețeaua N conform fig. BT4.2.4.(b); valoarea q reprezintă capacitatea resursei de stocare, adică numărul maxim de clienți ce pot aştepta.

Atragem atenția asupra faptului că într-o resursă de stocare partajată pot aştepta clienți de acelaşi tip, serviți pe mai multe linii (fluxuri) diferite, reprezentarea grafică a acestei situații constituind o generalizare firească a ilustrării din fig. BT4.2.4.(b). În această situație generală, reprezentată grafic în fig. BT4.2.4.(c) pentru clienți serviți pe k linii (fluxuri) diferite, funcționează următoarea teoremă.

Teorema 4.2.5. Fie $\boldsymbol{N'}$ reţeaua rezultată din $\boldsymbol{N}$ prin adăugarea poziţiei S_p şi a arcelor de conexiune, cu $M_0(S_p)= q$ (conform fig. BT4.2.4.(c)).

a) Dacă $\boldsymbol{N}$ este mărginită , atunci $\boldsymbol{N'}$ este mărginită.

b) Dacă $\boldsymbol{N}$ este viabilă, atunci $\boldsymbol{N'}$ este viabilă.

c) Dacă $\boldsymbol{N}$ este reversibilă, atunci $\boldsymbol{N'}$ este reversibilă.

Prin aplicarea repetată a **Teoremei 4.2.5**, la încheierea *Subpasului 2.2* al sintezei ascendente, toate resursele de stocare partajate (între două sau mai multe linii de servire) sunt ataşate modelului, conservând proprietăţile de mărginire, viabilitate şi reversibilitate.

BT4.2.2.4. Detalierea Pasului 3 (ataşarea resurselor specifice partajate)

Subpasul 3.1. Ataşarea resurselor specifice partajate paralel

Se consideră reţeaua $\boldsymbol{N}$ care include două poziţii O_1 şi O_2, modelând desfăşurarea în paralel a două operaţii ce necesită o aceeaşi resursă specifică, partajată paralel, R_{pp}. Modelarea utilizării partajate a lui R_{pp} se face printr-o poziţie cu marcajul iniţial $M_0(R_{pp})= 1$, conectată la reţeaua $\boldsymbol{N}$ conform fig. BT4.2.5.(a).

Într-o manieră cu totul similară, resursa specifică partajată poate fi utilizată de către două grupuri de operaţii desfăşurate în paralel, modelate prin două subreţele $\boldsymbol{N}_1$, $\boldsymbol{N}_2$ din $\boldsymbol{N}$. Modul de ataşare a lui R_{pp} în acest caz este ilustrat în fig. BT4.2.5.(b).

Reţeaua $\boldsymbol{N}$ include toate poziţiile de tip resursă (R_n, S_n, S_p) ataşate în *Pasul* 1 şi *Pasul* 2 al sintezei ascendente, deci şi pe cele ce servesc operaţiile din subreţelele $\boldsymbol{N}_1$ şi $\boldsymbol{N}_2$. Acest fapt este ilustrat generic în fig. BT4.2.5.(b) prin arcele trasate cu linie întreruptă.

Utilizarea corectă a lui R_{pp} necesită realizarea unei *structuri de excludere mutuală paralelă*, caracterizată prin:

- Ambele operaţii (grupuri de operaţii) desfăşurate în paralel au drepturi egale de a aloca R_{pp}.
- După alocarea lui R_{pp} la una din cele două operaţii (la unul din cele două grupuri de operaţii), operaţia (grupul de operaţii) se desfăşoară efectiv, permiţând, în final, eliberarea lui R_{pp}.

Cu alte cuvinte, nu se fac alocări ale lui R_{pp} care nu pot garanta şi eliberarea ei, datorită nedisponibilităţii altor resurse necesare desfăşurării operaţiei (grupului de operaţii) în cauză. Aceste caracteristici ale excluderii mutuale paralele (comentate anterior sub raport intuitiv) pot fi formalizate matematic prin următoarea definiţie.

Se spune că poziţia R_{pp}, împreună cu perechile de tranziţii $\{(t_{a1}, t_{b1}), (t_{a2}, t_{b2})\}$ existente în $\boldsymbol{N}$ (unde $R_{pp}^{\bullet} =\{t_{a1}, t_{a2}\}$ şi ${}^{\bullet}R_{pp}=\{t_{b1}, t_{b2}\}$, vezi fig. BT4.2.5.(b)) formează o excludere mutuală paralelă, dacă sunt îndeplinite următoarele condiţii:

1° Orice drum elementar dintre t_{ai} şi o poziţie asociată unei resurse generale trebuie să conţină t_{bi} (i= 1,2);

2° Orice circuit elementar ce conţine t_{ai} şi R_{pp} trebuie să conţină t_{bi} (i= 1,2);

3° Fiecare tranziţie de pe orice drum elementar dintre t_{ai} şi t_{bi} trebuie să aparţină şi unui drum elementar de operaţii (ce conţine numai poziţii asociate operaţiilor), unind t_{ai} cu t_{bi} (i= 1,2);

4° Oricare ar fi marcajul $M(\boldsymbol{N})$ care validează t_{ai}, dacă t_{ai} se execută, atunci există o secvenţă de executări de tranziţii din $\boldsymbol{N}_i$ care conduce la executarea lui t_{bi} (i= 1,2).

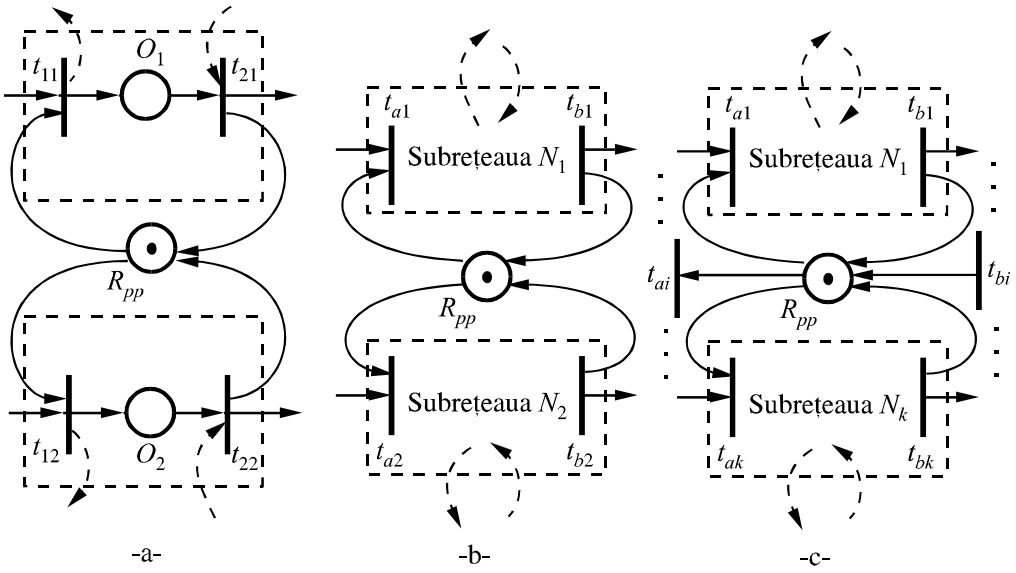


Fig. BT4.2.5. *Ilustrarea ataşării la rețeaua* $\boldsymbol{N}$ *a resursei specifice partajate paralel* R_{pp}
(a) R_{pp} *este utilizată de 2 operații* (*modelate prin* O_1 *şi* O_2);
(b) R_{pp} *este utilizată de 2 grupuri de operații* (*modelate prin subrețelele* $\boldsymbol{N}_1$ *şi* $\boldsymbol{N}_2$);
(c) R_{pp} *este utilizată de k grupuri de operații* (*modelate prin subrețelele* $\boldsymbol{N}_1$, $\boldsymbol{N}_2$,..., $\boldsymbol{N}_k$).

Evident condițiile de mai sus pot fi imediat generalizate pentru un număr arbitrar $k \geq 2$ de subrețele $\boldsymbol{N}_1$,..., $\boldsymbol{N}_k$ modelând k operații (grupuri de operații) desfăşurate în paralel, ce necesită o aceeaşi resursă specifică, partajată paralel, R_{pp}, conform fig. BT4.2.5.(c). În această situație, condițiile 1° – 4° ce definesc excluderea mutuală paralelă se vor referi la o mulțime de k perechi de tranziții existente în $\boldsymbol{N}$ $\{(t_{a1}, t_{b1}),..., (t_{ak}, t_{bk})\}$, unde $R_{pp}^{\bullet} = \{t_{a1},..., t_{ak}\}$ şi ${}^{\bullet}R_{pp} = \{t_{b1},..., t_{bk}\}$, adică precizarea $i = 1,2$ se înlocuieşte prin $i = 1,2,..., k$.

Teorema 4.2.6. Fie $\boldsymbol{N}'$ rețeaua ce rezultă din $\boldsymbol{N}$ prin adăugarea poziției R_{pp} şi a arcelor de conexiune, cu $M_0(R_{pp}) = 1$ (conform fig. BT4.2.5.(c)), astfel încât R_{pp} împreună cu perechile de tranziții $\{(t_{a1}, t_{b1}),..., (t_{ak}, t_{bk})\}$ formează o excludere mutuală paralelă.
a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci şi $\boldsymbol{N}'$ este mărginită (sigură).
b) Dacă $\boldsymbol{N}$ este viabilă, atunci şi $\boldsymbol{N}'$ este viabilă.
c) Dacă $\boldsymbol{N}$ este reversibilă, atunci şi $\boldsymbol{N}'$ este reversibilă.

Atragem atenția asupra generalității **Teoremei 4.2.6** care rămâne valabilă şi în cazul când una sau mai multe operații din subrețelele $\boldsymbol{N}_1$,..., $\boldsymbol{N}_k$ au deja ataşate alte resurse specifice cu utilizare partajată paralelă. În această situație, însă, satisfacerea condiției 4° din definirea excluderii mutuale secvențiale trebuie verificată cu deosebită atenție atunci când sunt ataşate toate resursele specifice partajate paralel, ale căror utilizări interferă pentru una sau mai multe operații.

Prin aplicarea repetată a **Teoremei 4.2.6**, la încheierea *Subpasului* 3.1 al sintezei ascendente, toate resursele specifice, partajate paralel, sunt ataşate modelului, conservând proprietățile de mărginire (siguranță), viabilitate şi reversibilitate.

Subpasul 3.2. Ataşarea resurselor specifice partajate secvențial

Se consideră rețeaua ***N*** care include două operații desfăşurate secvențial, ce necesită o aceeaşi resursă specifică, partajată secvențial, R_{ps}. Modelarea utilizării partajate a lui R_{ps} se face printr-o poziție cu marcajul inițial $M_0(R_{ps})= 1$, conectată la rețeaua ***N*** conform fig. BT4.2.6.(a). Într-o manieră cu totul similară, resursa specifică partajată poate fi utilizată de către două grupuri de operații desfăşurate secvențial, modelate prin două subrețele $\boldsymbol{N}_1$, $\boldsymbol{N}_2$ din ***N***. Modul de ataşare a lui R_{ps} în acest caz este ilustrat în fig. BT4.2.6.(b). Subrețeaua ***N*** include toate pozițiile de tip resursă (R_n, S_n, S_p, R_{pp}) ataşate în *Pasul* 1, *Pasul* 2 şi *Subpasul* 3.1 al sintezei ascendente, deci şi pe cele care servesc operațiile din subrețelele $\boldsymbol{N}_1$ şi $\boldsymbol{N}_2$. Acest fapt este ilustrat generic în fig. BT4.2.6.(b) prin arcele trasate cu linie întreruptă.

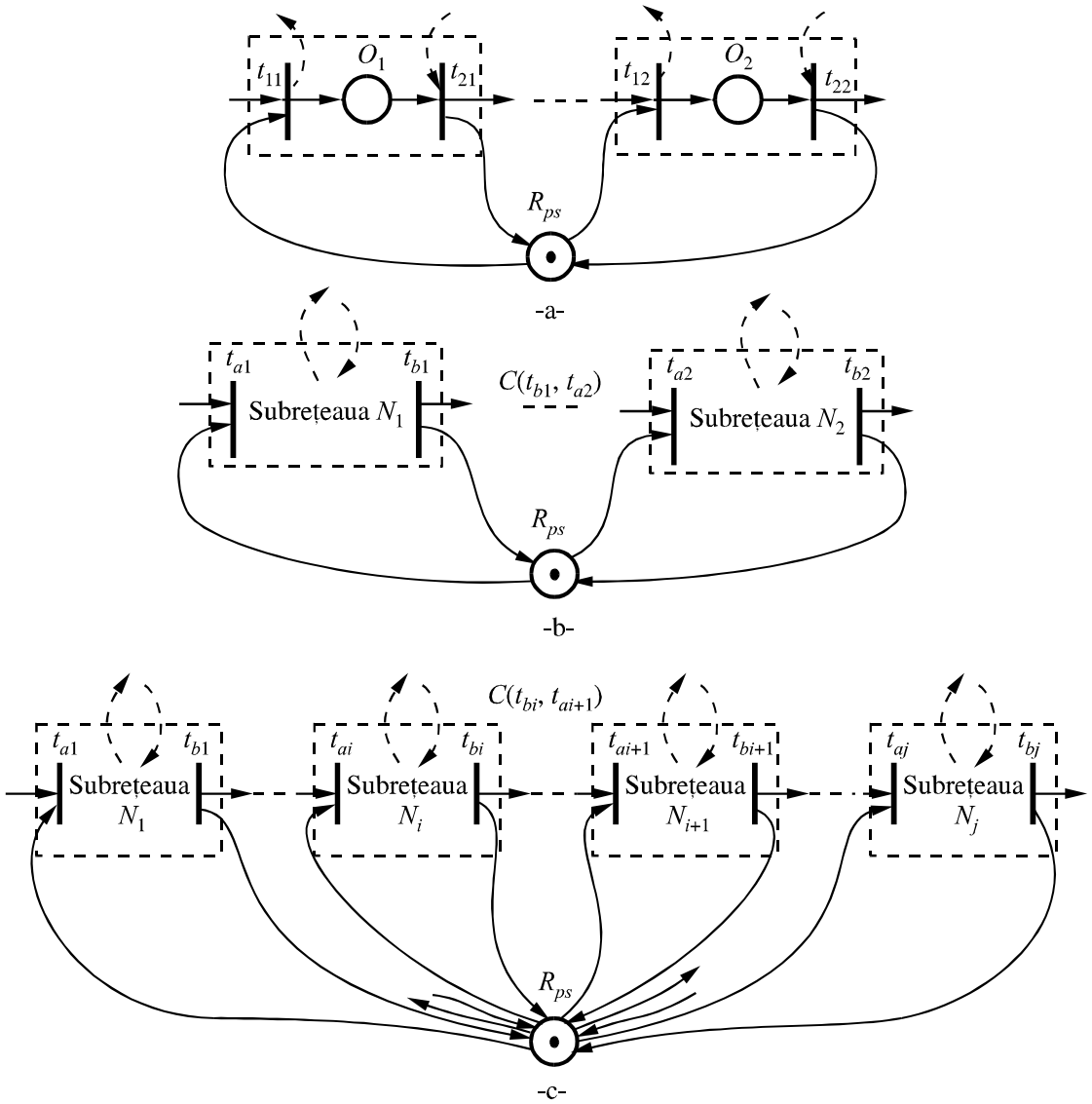


Fig. BT4.2.6. *Ilustrarea ataşării la rețeaua **N** a resursei specifice partajate secvențial* R_{ps}

(a) R_{ps} *este utilizată de 2 operații* (*modelate prin* O_1 *şi* O_2);

(b) R_{ps} *este utilizată de 2 grupuri de operații* (*modelate prin subrețelele* $\boldsymbol{N}_1$ *şi* $\boldsymbol{N}_2$);

(c) R_{ps} *este utilizată de j grupuri de operații* (*modelate prin subrețelele* $\boldsymbol{N}_1$, $\boldsymbol{N}_2$,.., $\boldsymbol{N}_j$).

Utilizarea corectă a lui R_{ps} necesită realizarea unei *structuri de excludere mutuală secvenţială*, caracterizată prin:

- Pentru un acelaşi client, operaţiile din $\boldsymbol{N}_1$ se desfăşoară înaintea operaţiilor din $\boldsymbol{N}_2$.
- După alocarea lui R_{ps} la una din cele două operaţii (grupuri de operaţii) desfăşurate secvenţial, operaţia (grupul de operaţii) se desfăşoară efectiv şi eliberează R_{ps}.

Aceste caracteristici ale excluderii mutuale secvenţiale (comentate anterior sub raport intuitiv) pot fi formalizate matematic prin următoarea definiţie.

Se spune că poziţia R_{ps}, împreună cu perechile de tranziţii $\{(t_{a1}, t_{b1}), (t_{a2}, t_{b2})\}$ existente în $\boldsymbol{N}$ (unde $R_{ps}^{\bullet} = \{t_{a1}, t_{a2}\}$ şi ${}^{\bullet}R_{ps} = \{t_{b1}, t_{b2}\}$, conform fig. BT4.2.6.(b)) formează o excludere mutuală secvenţială, dacă sunt îndeplinite următoarele condiţii:

1° Orice drum elementar dintre t_{ai} şi o poziţie asociată unei resurse generale trebuie să conţină şi t_{bi} (i= 1,2).

2° Orice circuit elementar ce conţine t_{ai} şi R_{ps} trebuie să conţină t_{bi} (i= 1,2).

3° Fiecare tranziţie de pe orice drum elementar dintre t_{ai} şi t_{bi} trebuie să aparţină şi unui drum elementar de operaţii (ce conţine numai poziţii asociate operaţiilor), unind t_{ai} cu t_{bi} (i= 1,2).

4° Pentru orice tranziţie $t \in \boldsymbol{N}_2$, există o tranziţie $t' \in \boldsymbol{N}_1$, astfel încât între t' şi t există un drum elementar de operaţii.

5° Nu există nici un drum elementar de operaţii de la $\boldsymbol{N}_2$ la $\boldsymbol{N}_1$.

6° Oricare ar fi marcajul $M(\boldsymbol{N})$ care validează t_{ai}, dacă t_{ai} se execută, atunci există o secvenţă de executări de tranziţii din $\boldsymbol{N}_i$ care conduce la executarea lui t_{bi} (i= 1,2).

Satisfacerea condiţiilor 1° – 5° se asigură printr-o topologie adecvată. În schimb, pentru îndeplinirea condiţiei 6° trebuie luat în considerare şi marcajul.

Satisfacerea condiţiei 6° se poate asigura prin următoarele *două metode*:

Metoda 1 – limitarea numărului total de clienţi (de acelaşi tip) admişi în sistem;

Metoda 2 – limitarea numărului de clienţi (de acelaşi tip) admişi pentru servire cu operaţiile ce se desfăşoară între utilizările resursei partajate secvenţial.

Detaliem, pe scurt, fiecare din cele două metode care sunt deopotrivă utilizate în practica proiectării.

Metoda 1 (*detaliere*)

Numărul de clienţi (de acelaşi tip) admişi în sistem este *limitat*, astfel încât, indiferent de etapele de servire în care se află fiecare client, resursa partajată să poată fi utilizată şi eliberată după utilizare. Limitarea se realizează prin fixarea *valorii maxime* a marcajului iniţial din poziţia corespunzătoare resursei generale.

Suportul matematic utilizat pentru determinarea acestei valori maxime îl constituie conceptul de *capacitate de jetoane*. Utilizând notaţiile din fig. BT4.2.6.(b), numim *capacitate de jetoane* între tranziţiile t_{b1} şi t_{a2}, notată prin $C(t_{b1}, t_{a2})$, numărul maxim de executări posibile ale lui t_{b1}, fără ca t_{a2} să se execute. Limitarea *marcajului iniţial al resursei generale* (ce serveşte atât $\boldsymbol{N}_1$ cât şi $\boldsymbol{N}_2$) la valoarea $C(t_{b1}, t_{a2})$ asigură satisfacerea condiţiei 6° (resursa generală nu este reprezentată în fig. BT4.2.6.(b)).

Justificarea acestei limitări este imediată, ținând cont de faptul că $C(t_{b1}, t_{a2})$ reprezintă numărul maxim de jetoane (adică clienți) care pot părăsi subrețeaua $\boldsymbol{N}_1$ (prin t_{b1}) fără a ajunge în subrețeaua $\boldsymbol{N}_2$ (prin t_{a2}). Deci, limitând numărul total de clienți admişi în sistem la valoarea $C(t_{b1}, t_{a2})$, în cazul cel mai nefavorabil de operare (adică pentru secvența de executări de tranziții cea mai nefavorabilă din punctul de vedere al condiției 6°), toți clienții sunt serviți numai cu operații corespunzătoare pozițiilor dintre t_{b1} şi t_{a2}. Astfel R_{ps} va fi în mod obligatoriu utilizată de $\boldsymbol{N}_2$, întrucât nu mai sunt clienți în sistem care să necesite R_{ps} pentru $\boldsymbol{N}_1$. Resursa partajată secvențial R_{ps} va fi solicitată de $\boldsymbol{N}_1$ numai după ce clientul cu stadiul cel mai avansat de servire va părăsi sistemul (deci implicit subrețeaua $\boldsymbol{N}_2$) şi va elibera o unitate a resursei generale, astfel încât să poată fi acceptat un nou client în sistem.

Metoda 2 (*detaliere*)

Numărul de clienți admişi pentru operațiile modelate prin pozițiile dintre t_{a1} şi t_{a2} este *limitat* la valoarea C (t_{b1}, t_{a2}), fără a limita numărul total de clienți (de acelaşi tip) admişi în sistem. Astfel, nu se mai impune nici o restricție asupra marcajului resursei generale. În această situație, cazul cel mai nefavorabil este analog celui semnalat anterior la *Metoda* 1, când toți cei C (t_{b1}, t_{a2}) clienți sunt serviți numai cu operații corespunzătoare pozițiilor dintre t_{b1} şi t_{a2}. Deci R_{ps} va fi în mod obligatoriu utilizată de $\boldsymbol{N}_2$, întrucât în $\boldsymbol{N}_1$ nu mai pot intra clienți.

Resursa partajată secvențial R_{ps} va fi solicitată de $\boldsymbol{N}_1$ numai după ce clientul cu stadiul cel mai avansat de servire dintre t_{b1} şi t_{a2} va ajunge în $\boldsymbol{N}_2$. Dar intrarea acestui client în $\boldsymbol{N}_2$ înseamnă că R_{ps} este deja alocată lui $\boldsymbol{N}_2$ (deci indisponibilă pentru solicitarea lui N_1), aşadar alocarea lui R_{ps} la $\boldsymbol{N}_1$ va fi posibilă numai după ce R_{ps} va fi eliberată de $\boldsymbol{N}_2$. Cu alte cuvinte executarea lui t_{a2} va fi urmată de o secvență de executări în $\boldsymbol{N}_2$ care conduce la executarea lui t_{b2}, condiția 6° fiind astfel satisfăcută.

Metoda 2 de satisfacere a condiției 6° poate fi realizată în mai multe *variante*, dintre care menționăm:

Varianta 1. Utilizarea unei structuri de control, denumită *kanban*, care să monitorizeze numărul de clienți acceptați pentru toate operațiile dintre t_{a1} şi t_{a2} şi să-l limiteze la valoarea $C(t_{b1}, t_{a1})$. (Termenul *kanban* este de origine japoneză şi are semnificația de „etichetă” sau „permis” pentru accesul unui client într-un modul al sistemului.)

Varianta 2. Alocarea *cu anticipare* în t_{a1} a resursei necesitate de prima operație de după t_{b1}. (Alocarea fără anticipare a acestei resurse s-ar face în t_{b1}).

Evident, condițiile de mai sus pot fi imediat generalizate pentru un număr arbitrar $j \geq 2$ de subrețele $\boldsymbol{N}_1,..., \boldsymbol{N}_j$ modelând j operații (grupuri de operații) desfăşurate secvențial, ce necesită o aceeaşi resursă specifică, partajată secvențial, R_{ps}, conform fig. BT4.2.6.(c). În această situație condițiile 1°, 2°, 3°, 6° ce definesc excluderea mutuală secvențială se vor referi la o mulțime de perechi de tranziții existente în $\boldsymbol{N}$ $\{(t_{a1}, t_{b1}),..., (t_{aj}, t_{bj})\}$, unde $R_{ps}^{\bullet} = \{t_{a1},...,t_{aj}\}$ şi ${}^{\bullet}R_{ps} = \{t_{b1},...,t_{bj}\}$, adică precizarea $i = 1,2$ se înlocuieşte prin $i = 1,2,...,j$. Pentru maximă claritate formulăm şi generalizarea condițiilor 4° , 5° în cazul celor j subrețele $\boldsymbol{N}_1,..., \boldsymbol{N}_j$.

Generalizarea condiției 4°. Pentru orice tranziție $t \in \boldsymbol{N}_{i+1}$, $i = 1,..., j-1$, există o tranziție $t' \in \boldsymbol{N}_i$, astfel încât între t' şi t există un drum elementar de operații.

Generalizarea condiției 5°. Nu există nici un drum elementar de operații de la $\boldsymbol{N}_{i+1}$ la $\boldsymbol{N}_i$, $i = 1,..., j-1$.

De asemenea se generalizează, cu uşurință, cele două metode prezentate anterior pentru satisfacerea condiției 6° din definiția excluderii mutuale secvențiale.

Generalizarea Metodei 1 revine la limitarea numărului total de clienți (de acelaşi tip), admişi în sistem, la valoarea $\min_{i=1,...,j-1} C(t_{bi}, t_{ai+1})$.

Generalizarea Metodei 2 revine la limitarea numărului de clienți admişi pentru operațiile desfăşurate între t_{ai} şi t_{ai+1} (adică între două utilizări consecutive ale resursei partajate secvențial R_{ps}) la valoarea $C(t_{bi}, t_{ai+1})$, pentru fiecare $i = 1,..., j-1$.

Teorema 4.2.7. Fie $\boldsymbol{N}'$ rețeaua ce rezultă din $\boldsymbol{N}$ prin adăugarea poziției R_{ps} şi a arcelor de conexiune, cu $M_0(R_{ps}) = 1$ (conform fig. BT4.2.6.(c)), astfel încât R_{ps} împreună cu perechile de tranziții $\{(t_{a1}, t_{b1}),..., (t_{aj}, t_{bj})\}$ formează o excludere mutuală secvențială.

a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci şi $\boldsymbol{N}'$ este mărginită (sigură).

b) Dacă $\boldsymbol{N}$ este viabilă, atunci şi $\boldsymbol{N}'$ este viabilă.

c) Dacă $\boldsymbol{N}$ este reversibilă, atunci şi $\boldsymbol{N}'$ este reversibilă.

Atragem atenția asupra faptului că **Teorema 4.2.7** rămâne valabilă şi în cazul când una sau mai multe operații din subrețelele $\boldsymbol{N}_1,..., \boldsymbol{N}_j$ au deja ataşate alte resurse specifice cu utilizare partajată secvențială. În această situație, însă, satisfacerea condiției 6° din definiția excluderii mutuale secvențiale trebuie verificată atunci când sunt ataşate toate resursele specifice partajate secvențial, ale căror utilizări interferă pentru una sau mai multe operații.

Prin aplicarea repetată a **Teoremei 4.2.7**, la încheierea *Subpasului* 3.2 al sintezei ascendente, toate resursele specifice, partajate secvențial, sunt ataşate modelului, conservând proprietățile de mărginire (siguranță), viabilitate şi reversibilitate.

Subpasul 3.3. Ataşarea resurselor specifice partajate paralel şi secvențial

Se consideră rețeaua $\boldsymbol{N}$ care include trei poziții O_1, O_2 şi O_3 modelând trei operații, dintre care O_1, O_2 se desfăşoară secvențial, dar ambele în paralel cu O_3, partajând resursa specifică R_{pg}. Resursa R_{pg} este *partajată generalizat*, adică, pe de o parte, *secvențial* de către O_1 şi O_2, iar pe de altă parte, *paralel* de O_3, în raport cu O_1 şi O_2. Modelarea utilizării partajate a lui R_{pg} se face printr-o poziție cu marcajul inițial $M_0(R_{pg}) = 1$, conectată la rețeaua N conform fig. BT4.2.7.(a). Într-o manieră cu totul similară, resursa specifică partajată R_{pg} poate fi utilizată de către trei grupuri de operații, dintre care două se desfăşoară secvențial (modelate prin subrețelele $\boldsymbol{N}_1$, $\boldsymbol{N}_2$ din $\boldsymbol{N}$), dar în paralel cu al treilea grup (modelat prin subrețeaua $\boldsymbol{N}_3$ din $\boldsymbol{N}$). Modul de ataşare a lui R_{pg} în acest caz este ilustrat în fig. BT4.2.7.(b). Subrețeaua $\boldsymbol{N}$ include toate pozițiile de tip resursă (R_n, S_n, S_p, R_{pp}, R_{ps}) ataşate în *Pasul* 1, *Pasul* 2, *Subpasul* 3.1 şi *Subpasul* 3.2 al sintezei ascendente, deci şi pe cele ce servesc operațiile din subrețelele $\boldsymbol{N}_1$, $\boldsymbol{N}_2$ şi $\boldsymbol{N}_3$. Acest fapt este ilustrat generic în fig. BT4.2.7.(b) prin arcele trasate cu linie întreruptă.

Se presupune că poziția R_{pg} împreună cu perechile de tranziții $\{(t_{a1}, t_{b1}), (t_{a2}, t_{b2}), (t_{a3}, t_{b3})\}$ existente în $\boldsymbol{N}$ (unde $R_{pg}^{\bullet}=\{t_{a1}, t_{a2}, t_{a3}\}$ şi ${}^{\bullet}R_{pg}=\{t_{b1}, t_{b2}, t_{b3}\}$, conform fig. BT4.2.7.(b)) formează o *excludere mutuală generalizată*, dacă sunt îndeplinite următoarele condiții:

1° Structurile R_{pg} împreună cu $\{(t_{a1}, t_{b1}), (t_{a3}, t_{b3})\}$ şi R_{pg} împreună cu $\{(t_{a2}, t_{b2}), (t_{a3}, t_{b3})\}$ sunt excluderi mutuale paralele.

2° Structura R_{pg} împreună cu $\{(t_{a1}, t_{b1}), (t_{a3}, t_{b3})\}$ este o excludere mutuală secvențială.

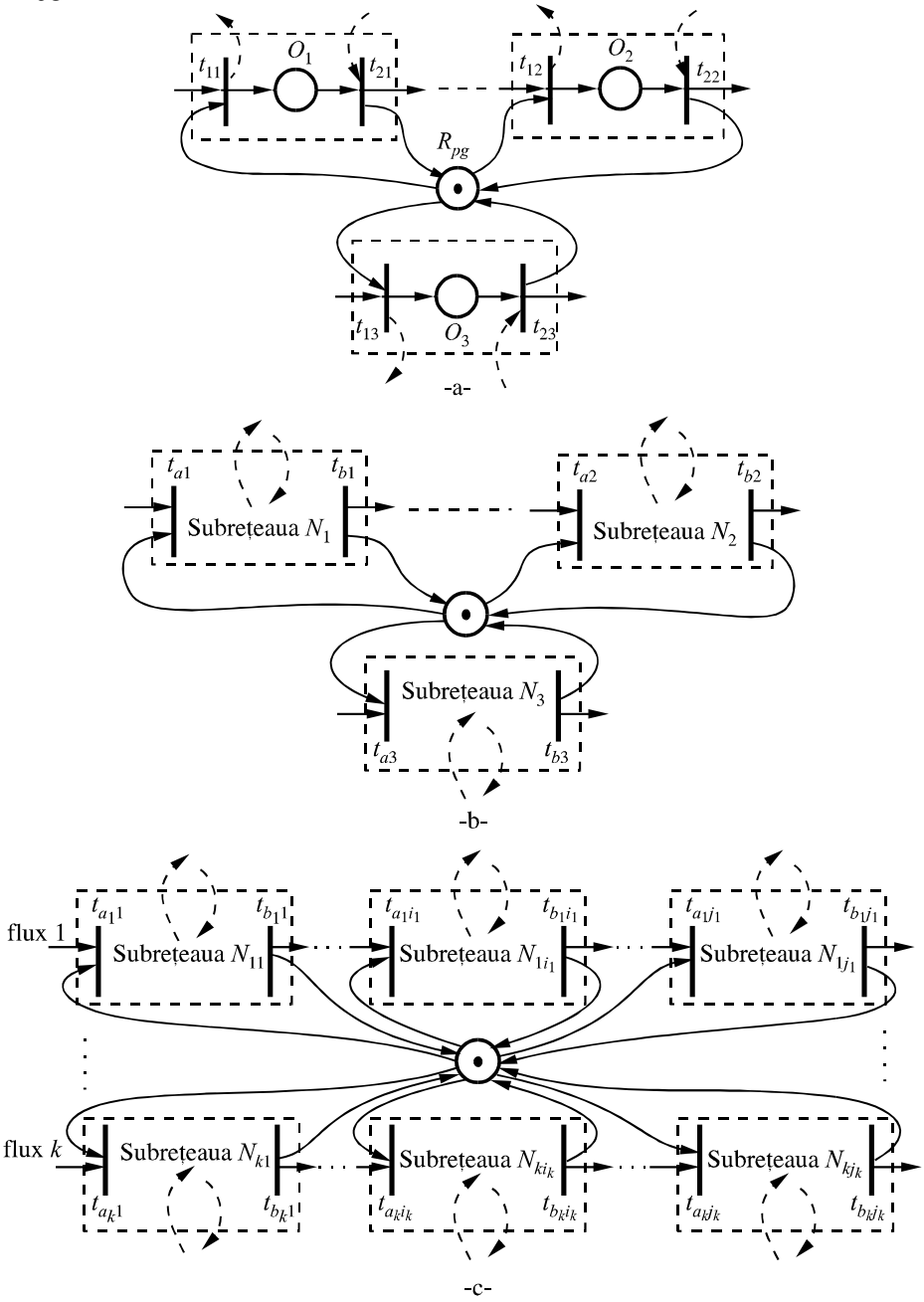


Fig. BT4.2.7. *Ilustrarea ataşării la rețeaua* $\boldsymbol{N}$ *a resursei specifice partajate generalizat* R_{pg}

(a) R_{pg} *este utilizată de 3 operații* (*modelate prin* O_1, O_2 *şi* O_3);

(b) R_{pg} *este utilizată de 3 grupuri de operații* (*modelate prin* $\boldsymbol{N}_1$, $\boldsymbol{N}_2$ *şi* $\boldsymbol{N}_3$);

(c) R_{pg} *este utilizată de* $j_1+...+j_k$ *grupuri de operații* (*modelate prin* $\boldsymbol{N}_{11}$,..., $\boldsymbol{N}_{1j_1}$;...; $\boldsymbol{N}_{k1}$... $\boldsymbol{N}_{kj_k}$).

Condițiile de mai sus pot fi imediat extinse pentru un număr arbitrar de subrețele, care partajează generalizat o resursă specifică R_{pg}, conform fig. BT4.2.7.(c), unde excluderea mutuală secvențială se realizează între subrețele situate de-a lungul fiecăreia din cele k căi (drumuri, fluxuri) de operații, iar excluderea mutuală paralelă se realizează între subrețele situate pe căi diferite. Nu vom dezvolta o formulare matematică riguroasă pentru acest caz extins, datorită complexității descrierii, dar considerăm că reprezentarea grafică din fig. BT4.2.7.(c) constituie un suport intuitiv suficient pentru a înțelege specificul excluderii mutuale generalizate.

Teorema 4.2.8. Fie $\boldsymbol{N'}$ rețeaua ce rezultă din $\boldsymbol{N}$ prin adăugarea poziției R_{pg} şi a arcelor de conexiune, cu $M_0(R_{pg}) = 1$ (conform fig. BT4.2.7.(c)), astfel încât R_{pg} împreună cu perechile de tranziții $\{(t_{a_11}, t_{b_11}),..., (t_{a_1j_1}, t_{b_1j_1});...; (t_{a_k1}, t_{b_k1}),..., (t_{a_kj_k}, t_{b_kj_k})\}$ formează o excludere mutuală generalizată.

a) Dacă $\boldsymbol{N}$ este mărginită (sigură), atunci şi $\boldsymbol{N'}$ este mărginită (sigură).

b) Dacă $\boldsymbol{N}$ este viabilă, atunci şi $\boldsymbol{N'}$ este viabilă.

c) Dacă $\boldsymbol{N}$ este reversibilă, atunci şi $\boldsymbol{N'}$ este reversibilă.

Precizăm faptul că **Teorema 4.2.8** rămâne valabilă şi în cazul când una sau mai multe operații din subrețelele $\boldsymbol{N}_{11},..., \boldsymbol{N}_{1j_1};..., \boldsymbol{N}_{kj_k}$ au deja ataşate alte resurse specifice partajate generalizat, pentru care a fost asigurată excluderea mutuală generalizată.

Prin aplicarea repetată a **Teoremei 4.2.8**, la încheierea *Subpasului* 3.2 al sintezei ascendente, toate resursele specifice, partajate generalizat sunt ataşate modelului, conservând proprietățile de mărginire (siguranță), viabilitate şi reversibilitate.

Încheiem această secțiune dedicată sintezei hibride subliniind faptul că, deşi detalierea operațiilor şi ataşarea resurselor se realizează în două etape distincte, proiectantul trebuie să aibă în vedere *corelarea lor pe tot parcursul sintezei*, pentru a evita introducerea unor erori de modelare.

BT4.3. Proprietăți caracteristice ale structurilor de conducere rezultate din sinteză

Modelul de tip rețea Petri al structurii de conducere capabil să descrie *atât dinamica operațiilor cât şi a resurselor* prezintă următoarele caracteristici topologice şi de marcaj.

- Pozițiile corespunzătoare fiecărei succesiuni distincte de operații (linie sau flux de operații) sunt secvențiate de-a lungul unei *căi* (drum) *de operații*, asociate succesiunii respective. În cazul când există alegeri, concurențe sau sincronizări între operații, într-o cale de operații pot apărea mai multe *subcăi de operații*.
- Tranzițiile sunt situate *numai* pe căile de operații şi, eventual pe subcăile acestora, marcând sfârşitul unei operații şi începutul celei ce urmează, în conformitate cu succesiunile de servicii ce trebuie prestate clienților.
- Pozițiile corespunzătoare resurselor se conectează la tranzițiile de pe căile şi subcăile de operații *fără a introduce* tranziții suplimentare.

- În absența unor solicitări anume, o resursă *specifică* se alocă la începutul operației pe care o deserveşte şi se eliberează la sfârşitul operației respective (alocare fără anticipare).
- În absența altor precizări, o resursă *generală* se alocă la începutul servirii unui client şi se eliberează după servirea completă a clientului.
- Marcajul inițial lasă vide toate pozițiile corespunzătoare *operațiilor* şi plasează jetoane numai în pozițiile corespunzătoare *resurselor*.

BT4.4. Funcțiuni de bază şi considerații de proiectare ale controlerului procedural

BT4.4.1. Funcțiuni de bază

Controlerul secvențial rezultă *direct din rețeaua Petri a structurii de conducere* (proces + controler) obținută prin aplicarea procedurii de sinteză:

- Pozițiile acestei rețele modelează *starea* operațiilor şi resurselor procesului condus.
- Conexiunile pozițiilor la tranziții *definesc controlerul secvențial printr-un set de reguli*, fiecărei tranziții corespunzându-i o regulă de forma IF *precondiții* THEN *postcondiții*.

Setul de reguli IF THEN ce alcătuiesc controlerul permite *conducerea procesului în conformitate cu dinamica rețelei rezultate prin sinteză*. Cu alte cuvinte, controlerul asigură fizic *funcționarea mecanismului de validare şi executare a tranzițiilor*, discutat din punct de vedere teoretic în paragraful BT2.3 din capitolul 2. Implementarea setului de reguli IF THEN ale controlerului se poate realiza hard sau soft. Astfel, rețeaua Petri rezultată din sinteză poate fi privită prin prisma schemei bloc din fig. BT4.1.1, concluzionând faptul că blocul "Proces de servire" conține toate pozițiile rețelei, iar blocul "Controler" conține toate tranzițiile. Acest mod de înțelegere al structurilor de conducere a proceselor cu evenimente discrete corespunde punctului de vedere standard din automatică, conform căruia obiectul condus (ca entitate fizică a cărei dinamică trebuie controlată) este separat față de controler (ca mecanism ce implementează principiile de conducere, pe baza informațiilor primite de la obiectul condus).

BT4.4.2. Considerații de proiectare

Proiectarea unui controler, care să asigure îndeplinirea, de către structura de conducere, a condițiilor **(C1)** – **(C4)** precizate în paragraful BT4.1 al acestui capitol, se desfăşoară în două etape:

Etapa 1: Utilizând *drept date de proiectare* informațiile despre *disponibilul de resurse* şi modul cum acestea pot fi utilizate pentru realizarea *succesiunilor de operații* solicitate de toți clienții, se elaborează *un model tip rețea Petri al structurii de conducere*, prin aplicarea uneia dintre tehnicile de sinteză prezentate în secțiunile anterioare ale acestui capitol.

Etapa 2: Folosind rețeaua Petri rezultată din Etapa 1, se construieşte setul de reguli al controlerului, asociind fiecărei tranziții o *regulă* de forma:

IF (operația curentă este completă
and resursele pentru operația următoare sunt disponibile)
THEN (eliberează resursele folosite de operația curentă
and alocă noile resurse necesare *and* începe operația următoare).

În aplicațiile uzuale de conducere a proceselor pilotate de evenimente, satisfacerea condițiilor **(C1) – (C4)** necesitate de funcționarea corectă a structurii de conducere revine la sintetizarea, în Etapa 1, a unei rețele Petri ce prezintă următoarele proprietăți:

- *viabilitate* – are semnificația începerii şi sfârşirii tuturor operațiilor, atâta vreme cât există clienți care trebuie serviți;
- *mărginire* – are semnificația nedepăşirii capacităților de operare a resurselor;
- *reversibilitate* – are semnificația repetabilității serviciilor prestate pentru toți clienții de acelaşi tip sosiți în sistem.

Se constată, aşadar, că manipularea eficientă a tehnicilor de sinteză reprezintă o condiție fundamentală pentru construirea setului de reguli ale controlerului secvențial. În plus, proprietățile comportamentale şi structurale ale rețelei Petri rezultate din sinteză, trebuie probate prin tehnici de analiză adecvate, astfel încât dinamica structurii de conducere să fie cunoscută într-o manieră cât se poate de completă. Din acest motiv, în Etapa 1 de proiectare a controlerului, proiectantul trebuie să facă uz de întreaga sa experiență în utilizarea formalismului rețelelor Petri (sinteză şi analiză). În cazul rețelelor cu o topologie relativ complexă, simularea şi analiza asistată de calculator reprezintă un instrument indispensabil unei investigații detaliate.

Atragem atenția asupra următorului fapt important. Strategia de proiectare *în două etape* a controlerului coincide, ca mod de organizare, cu filosofia tipică din cazul sistemelor pilotate de timp, care, de asemenea, cuprinde două faze:

Faza 1 – Propunerea unui model (în circuit închis) pentru structura de conducere (proces + controler) care să asigure dinamica dorită pentru proces.

Faza 2 – Utilizând modelul în circuit închis, construit în Faza 1 şi cunoştințele despre funcționarea procesului, se formulează principiile de operare ale controlerului.

Deosebirea fundamentală constă în aceea că, pentru sistemele cu evenimente discrete, cerințele impuse mai sus funcționării structurii de conducere sunt de factură logică, iar proiectarea se bazează pe o abordare calitativă, în timp ce pentru sistemele pilotate de timp, se impun, uzual, şi performanțe temporale, proiectarea necesitând, astfel, şi o abordare cantitativă.

Aplicaţii

AP4.1.

Se doreşte sintetizarea unei structuri de conducere pentru un sistem de calcul biprocesor cu reprezentarea schematică din fig. AP3.5.1 şi principiile de funcționare prezentate în **AP3.5**. Se urmăreşte construirea unui model tip rețea Petri netemporizată cu proprietățile de siguranță, viabilitate şi reversibilitate.

1. Să se aplice algoritmul de sinteză hibridă. Pentru fiecare pas al algoritmului se va prezenta reţeaua Petri precizând semnificaţia fiecărei poziţii nou introduse în raport cu pasul precedent.
2. Să se verifice prin simulare în mediul ***Petri Net Toolbox***, pentru 10.000 de evenimente, funcţionarea corectă a structurii de conducere şi să se determine:
   i) numărul de fişiere create iniţial pe D_1 şi copiate pe D_2;
   ii) numărul de fişiere create iniţial pe D_2 şi copiate ulterior pe D_1.

Soluţie

1. Modelul de tip reţea Petri corespunzător succesiunilor de operaţii, rezultat prin rafinare descendentă, este prezentat în fig. AP4.1.1. Există două *căi de operaţii* (corespunzătoare fiecăruia din cele două tipuri de taskuri) la care se conectează cele două resurse generale (şi anume procesoarele).

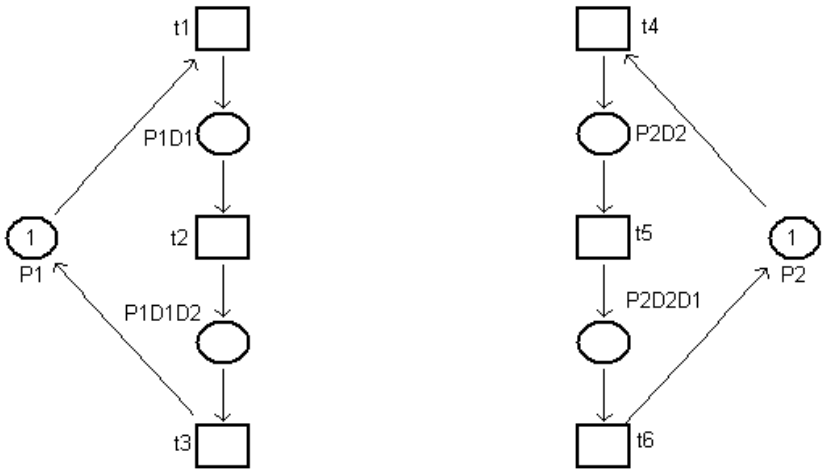


Fig. AP4.1.1. *Modelarea operaţiilor în etapa de sinteză descendentă.*

Acestui model i se ataşează resursele prin sinteză ascendentă.

Pasul 1. În sistem nu există resurse specifice nepartajate astfel că trecem la pasul următor.

Pasul 2. Deoarece sistemul de calcul nu conţine resurse de stocare, se trece la pasul următor.

Pasul 3. Ataşăm resursele specifice partajate.

Subpasul 3.1. Ataşăm resursele partajate paralel.

Introducem resursele partajate paralel D_1 şi D_2. Deoarece prin includerea acestor resurse în model se poate pierde proprietatea de viabilitate a reţelei (vezi aplicaţia **AP3.3**), vom asigura excluderea mutuală paralelă, alocând cu anticipare discul D_2 la începerea unui task ST_1. Structura rezultată (fig. AP4.1.2) posedă proprietăţile de viabilitate, mărginire şi reversibilitate.

Resursele necesare sistemului de calcul fiind introduse în totalitate, nu mai este cazul executării *Subpaşilor 3.2* şi *3.3*, modelul obţinut după aplicarea *Subpasului 3.1* fiind modelul final.

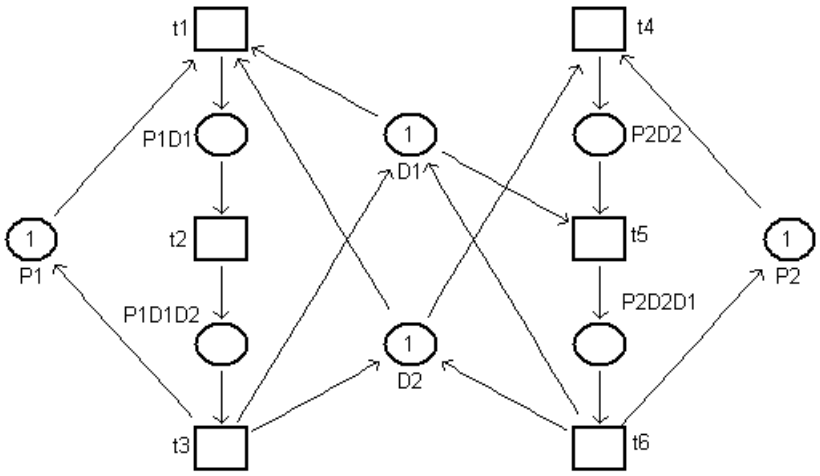


Fig. AP4.1.2. *Modelul rezultat în urma aplicării algoritmului de sinteză hibridă*

2. Utilizând modul de simulare ***Run Fast***, pentru 10.000 de evenimente se obțin indicatorii globali de performanță prezentați în fig. AP4.1.3. Numărul de fișiere create inițial pe D_1 și copiate pe D_2 corespunde indicatorului ***Service Sum*** pentru tranziția t_3, și anume 1672, iar numărul de fișiere create inițial pe D_2 și copiate ulterior pe D_1 corespunde aceluiași indicator pentru tranziția t_6, având valoarea 1661. Cititorul este invitat să formuleze o soluție bazată pe consultarea indicatorilor globali de performanță referitori la pozițiile din rețea.

Petri Net Toolbox - Web Browser

Global Statistics: Transitions

Model:cap4pr1_pas2.xml

Events:10000

Time:0

Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	1673	0	0	0	0
t2	1672	0	0	0	0
t3	1672	0	0	0	0
t4	1661	0	0	0	0
t5	1661	0	0	0	0
t6	1661	0	0	0	0

Ok Save

Fig. AP4.1.3. *Indicatorii globali de performanță obținuți în urma simulării în mediul* ***Petri Net Toolbox*** *pentru un număr de 10.000 evenimente.*

AP4.2.

Se doreşte sintetizarea unei structuri de conducere pentru un sistem de fabricaţie cu reprezentarea schematică din fig. AP3.6.1 şi principiile de funcţionare prezentate în **AP3.6**. Se urmăreşte construirea unui model tip reţea Petri netemporizată cu proprietăţile de mărginire, viabilitate şi reversibilitate.

1. Să se aplice algoritmul de sinteză hibridă pentru următoarele două situaţii:
   (i) Nici o resursă nu se alocă înainte de a se începe operaţia care necesită resursa respectivă, iar numărul de palete x este stabilit la valoarea maximă care permite îndeplinirea condiţiilor **(C1)** – **(C4)** impuse funcţionării structurii de conducere.
   (ii) Resursele pot fi alocate şi cu anticipare (înainte de a se începe operaţia care le necesită efectiv), iar numărul de palete este $x = 4$.

   Pentru fiecare pas al algoritmului se va prezenta reţeaua Petri precizând semnificaţia fiecărei poziţii nou introduse în raport cu pasul precedent
2. Să se verifice prin simulare în mediul ***Petri Net Toolbox***, pentru 10.000 de evenimente, funcţionarea corectă a celor două structuri de conducere construite la punctul 1 şi să se determine numărul total de piese prelucrate în fiecare din aceste cazuri.

Soluţie

1. Succesiunea de operaţii solicitată de clienţi (piese) precum şi resursa generală (care în cazul de faţă este reprezentată de palete) modelată prin poziţia P, este prezentată în fig. AP4.2.1. Reţeaua Petri, rezultată prin rafinare descendentă, posedă proprietăţile de viabilitate, mărginire şi reversibilitate.

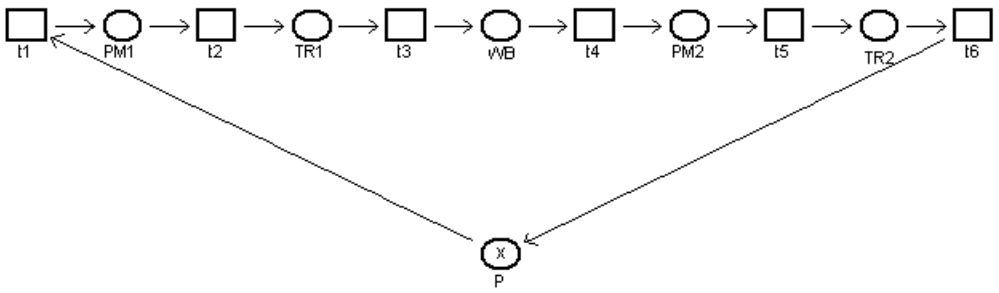


Fig. AP4.2.1. *Modelul rezultat după etapa descendentă a sintezei hibride.*

Acestui model i se ataşează resursele prin sinteză ascendentă.

Pasul 1: Ataşăm resursele specifice nepartajate: maşina M_1 (modelată prin poziţia M1) şi maşina M_2 (modelată prin poziţia M2). Reţeaua obţinută în urma pasului 1 al sintezei ascendente este reprezentată în fig. AP.4.2.2.

Pasul 2: Ataşăm resursele de stocare.

Subpasul 2.1. Ataşăm resursa de stocare nepartajată D (modelată prin poziţia D) rezultând reţeaua Petri din fig. AP4.2.3.

Subpasul 2.2. Nu există resurse de stocare partajate.

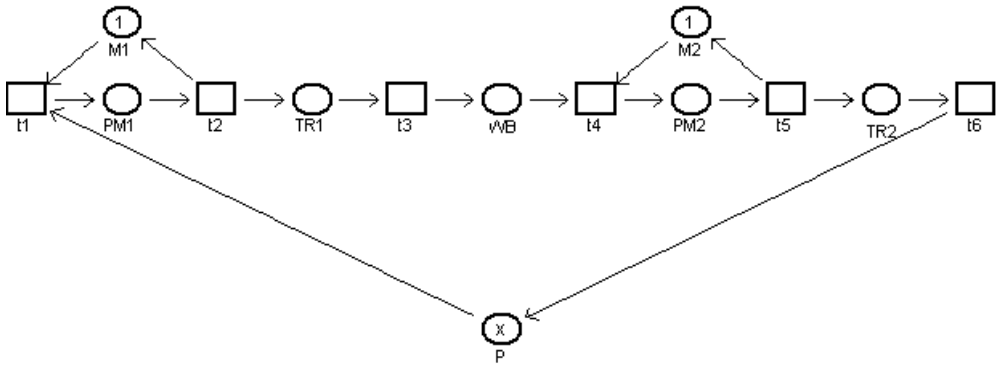


Fig. AP4.2.2. *Modelul rezultat după aplicarea Pasului 1 al etapei de sinteză ascendentă.*

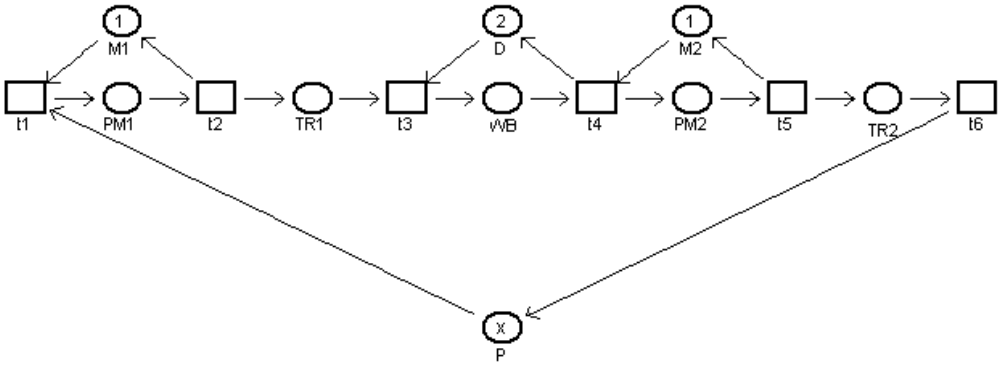


Fig. AP4.2.3. *Modelul rezultat după aplicarea Pasului 2 al etapei de sinteză ascendentă.*

Pasul 3: Ataşăm resursele specifice partajate.

Subpasul 3.1. Nu există resurse specifice partajate paralel.

Subpasul 3.2. Ataşăm resursa partajată secvențial (robotul) reprezentată prin poziția R. Pentru ca R împreună cu perechile de tranziții $\{(t_2, t_3), (t_5, t_6)\}$ să formeze o excludere mutuală secvențială aplicăm următoarele două metode corespunzătoare cazurilor (i) şi (ii) din enunț.

Metoda 1 limitează numărul total de clienți prin utilizarea unui număr maxim de $x = 3$ palete. Rețeaua rezultată este prezentată în fig. AP4.2.4 şi este viabilă, mărginită şi reversibilă.

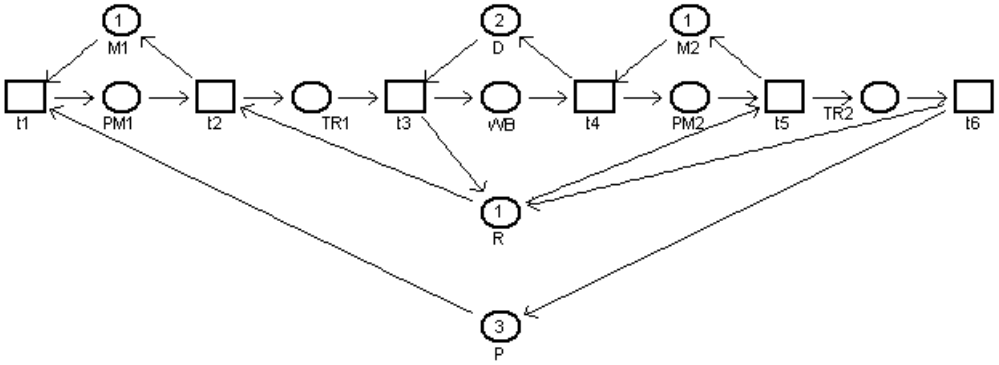


Fig. AP4.2.4. *Modelul rezultat după aplicarea Pasului 3 al etapei de sinteză ascendentă, cu limitarea numărului total de clienți admişi în sistem prin intermediul numărului de palete.*

Metoda 2 se bazează pe alocarea cu anticipare (în t_2) a unui loc din depozitul D. Rețeaua rezultată este prezentată în fig. AP4.2.5 şi este viabilă, mărginită şi reversibilă.

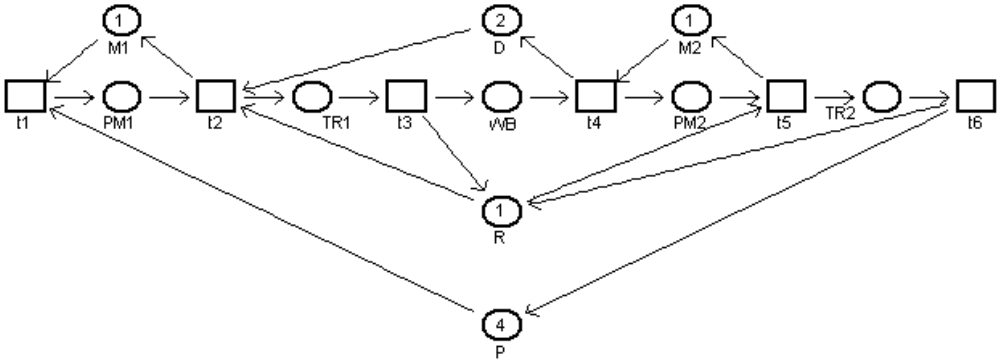
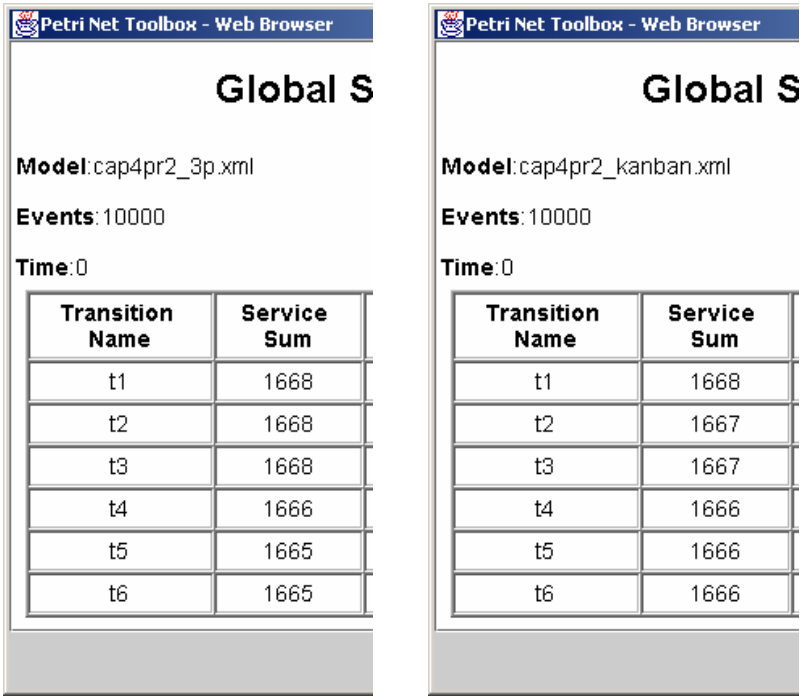


Fig. AP4.2.5. *Modelul rezultat după Pasul 3 al etapei de sinteză ascendentă prin alocarea cu anticipare a unui loc din depozitul D.*

În sistem nu avem resurse partajate concomitent paralel şi secvențial, în consecință modelul rezultat după *Subpasul 3.2* este cel final.

2. Simulând ambele modele în mediul ***Petri Net Toolbox*** pentru 10.000 de evenimente şi utilizând indicatorii globali de performanță corespunzători tranzițiilor, putem obține numărul total de piese prelucrate în sistem prin consultarea indicatorului ***Service Sum*** pentru tranziția t_6, conform fig. AP4.2.6. Cititorul este invitat să formuleze o soluție bazată pe consultarea indicatorilor globali de performanță referitori la pozițiile din cele două rețele.



(a) (b)

Fig. AP4.2.6. *Indicatorul **Service Sum** pentru tranzițiile rețelelor din* (a) *fig. AP4.2.4 şi* (b) *fig. AP4.2.5.*

AP4.3.

Se consideră o celulă de fabricație flexibilă, adaptată după (Zhou and DiCesare, 1993), posedând următoarele resurse, amplasate conform reprezentării schematizate din fig. AP4.3.1.

- cinci maşini cu comandă numerică M_1, M_2, M_3, M_4, M_5;
- două transportoare T_1, T_2;
- patru roboți R_1, R_2, R_3, R_4;
- două depozite interoperaționale D_1, D_2 de capacități 4 şi respectiv 3.

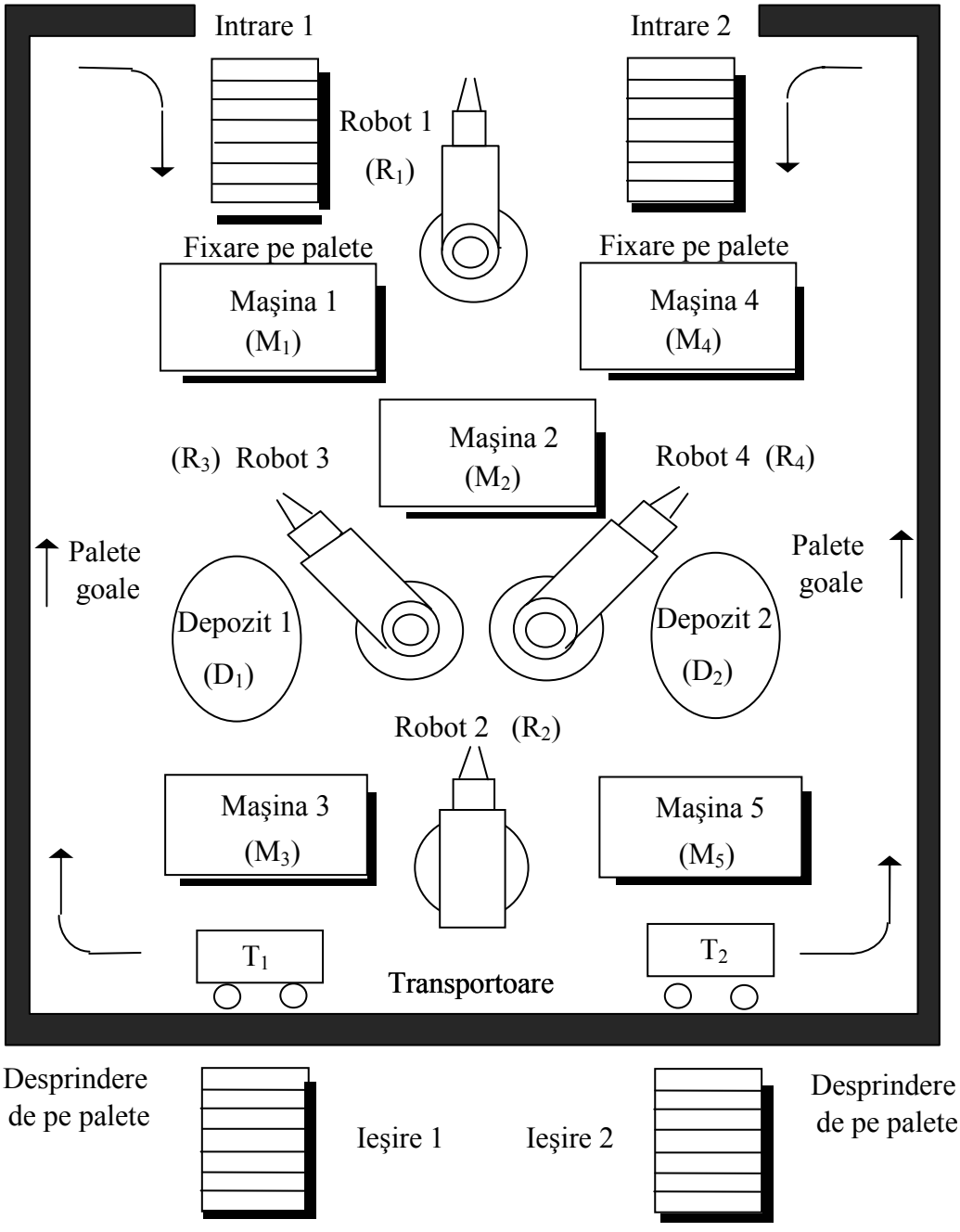


Fig. AP4.3.1. *Reprezentarea schematizată a celulei flexibile de fabricație.*

La cele două intrări se prezintă două tipuri de piese brute, PB_1 şi PB_2, dintr-o magazie ce conţine un număr suficient de mare de piese brute de ambele tipuri.

Fluxul tehnologic de prelucrare a pieselor de tipul PB_1 constă în următoarea succesiune de operaţii, care produce piese finite de tipul PF_1, livrate la ieşirea 1:

1^0 Piesa fixată pe o paletă este încărcată de către R_1 pe M_1 sau M_2.
2^0 Piesa este prelucrată fie pe M_1, fie pe M_2 (executându-se operaţii identice).
3^0 Piesa este descărcată de pe M_1 sau M_2 cu ajutorul lui R_3 în depozitul D_1.
4^0 Piesa aşteaptă în D_1.
5^0 Piesa este luată din D_1 de către R_3 şi încărcată pe M_3.
6^0 Piesa este prelucrată pe M_3.
7^0 Piesa este descărcată de pe M_3 cu ajutorul lui R_2 pe transportorul T_1.
8^0 Piesa este transportată de T_1 la ieşire.

La ieşirea 1, PF_1 este desprinsă de pe paletă, iar paleta goală este trimisă la intrarea 1.

Fluxul tehnologic de prelucrare a pieselor de tipul PB_2 constă în următoarea succesiune de operaţii, care produce piese de tipul PF_2, livrate la ieşirea 2.

1^0 Piesa fixată pe o paletă este încărcată de către R_1 pe M_4.
2^0 Piesa este prelucrată pe M_4.
3^0 Piesa este descărcată de pe M_4 cu ajutorul lui R_4 în depozitul D_2.
4^0 Piesa aşteaptă în D_2.
5^0 Piesa este luată din D_2 de către R_4 şi încărcată pe M_5.
6^0 Piesa este prelucrată pe M_5.
7^0 Piesa este descărcată de pe M_5 cu ajutorul lui R_2 pe transportorul T_2.
8^0 Piesa este transportată de T_2 la ieşire.

La ieşirea 2, PF_2 este desprinsă de pe paletă, iar paleta goală este trimisă la intrarea 2.

Se urmăreşte construirea unui model tip reţea Petri netemporizată cu proprietăţile de mărginire, viabilitate şi reversibilitate. Se presupune că nici o resursă nu se alocă cu anticipare, adică înainte de a începe operaţia care necesită efectiv resursa respectivă. Numărul de palete folosite pentru cele două fluxuri tehnologice, x_1 şi, respectiv x_2, este stabilit la valoarea maximă care permite îndeplinirea condiţiilor **(C1) – (C4)** impuse funcţionării structurii de conducere.

1. Să se aplice algoritmul de sinteză hibridă. Pentru fiecare pas al algoritmului se va reprezenta reţeaua Petri precizând semnificaţia fiecărei poziţii nou introduse în raport cu pasul precedent.
2. Să se verifice prin simulare în mediul ***Petri Net Toolbox***, pentru 10.000 de evenimente, funcţionarea corectă a structurii de conducere şi să se determine:
   (i) numărul de piese transportate de R_1;
   (ii) numărul de piese prelucrate pe M_1;
   (iii) numărul de piese prelucrate pe M_2;
   (iv) numărul de piese prelucrate pe M_4.
3. Să se scrie regulile IF THEN ale controlerului procedural care comandă:
   (i) alocarea robotului R_1 pentru începerea unei operaţii;
   (ii) eliberarea robotului R_1 după încheierea unei operaţii.

4. Să se comenteze funcționarea structurii de conducere în cazul în care se modifică numărul de palete utilizate astfel:
   (i) x_1-1, x_2;
   (ii) x_1, x_2-1;
   (iii) x_1+1, x_2;
   (iv) x_1, x_2+1;
   (v) x_1+1, x_2+1;
   unde valorile x_1, x_2 sunt cele rezultate din sinteza de la punctul 1. Pentru fiecare din situațiile de la subpunctele (i) – (v) se va preciza care dintre proprietățile de mărginire, viabilitate şi reversibilitate se păstrează şi care se pierd, explicând totodată motivul.
5. Să se arate că, indiferent de numărul de palete utilizate, există întotdeauna strategii de a stabili priorități în alocarea unei resurse partajate, astfel încât funcționarea structurii de conducere să îndeplinească condițiile **(C1) – (C4)** impuse funcționării structurii de conducere.
6. Să se simuleze în mediul ***Petri Net Toolbox***, pentru 10.000 de evenimente, funcționarea structurii de conducere cu $x_1 = x_2 = 6$ pentru cazul când alocarea unor resurse se realizează pe baza strategiilor de priorități studiate la punctul 5. Să se determine:
   (i) numărul de piese transportate de R_1;
   (ii) numărul de piese prelucrate de M_1;
   (iii) numărul de piese prelucrate de M_2;
   (iv) numărul de piese prelucrate de M_4.

Soluție

1. Modelul de tip rețea Petri corespunzător succesiunilor de operații şi resurselor generale, rafinat prin sinteză descendentă, este prezentat în fig. AP4.3.2. Există două *căi de operații* (corespunzătoare fiecăruia din cele două tipuri de piese) la care se conectează cele două resurse generale (P1 şi P2, care modelează paletele disponibile pentru fiecare tip de piese).

Semnificațiile pozițiilor sunt în concordanță cu operațiile care se desfăşoară pe cele două fluxuri din sistem: P1 – palete disponibile pentru piese de tipul PB_1; T1R1 – încărcarea pe M_1 sau M_2; PM1 – prelucrare pe M_1; PM2 – prelucrare pe M_2; T1R3 – descărcare de pe M_1 sau M_2 în D; WD1 – stocare în D_1; T2R3 – încărcare din D_1 pe M_3; PM3 – prelucrare pe M_3; T1R2 – descărcare de pe M_3 pe T_1; TT1 – transport cu T_1; P2 – palete disponibile pentru piese de tipul PB_2; T2R1 – încărcare pe M_4; PM4 – prelucrare pe M_4; T1R4 – descărcare de pe M_4 în D_2; WD2 – stocare în D_2; T2R4 – încărcare din D_2 pe M_5; PM5 – prelucrare pe M_5; T2R2 – descărcare de pe M_5 pe T_2; TT2 – transport cu T_2.

Acestui model i se ataşează resursele prin sinteză ascendentă.

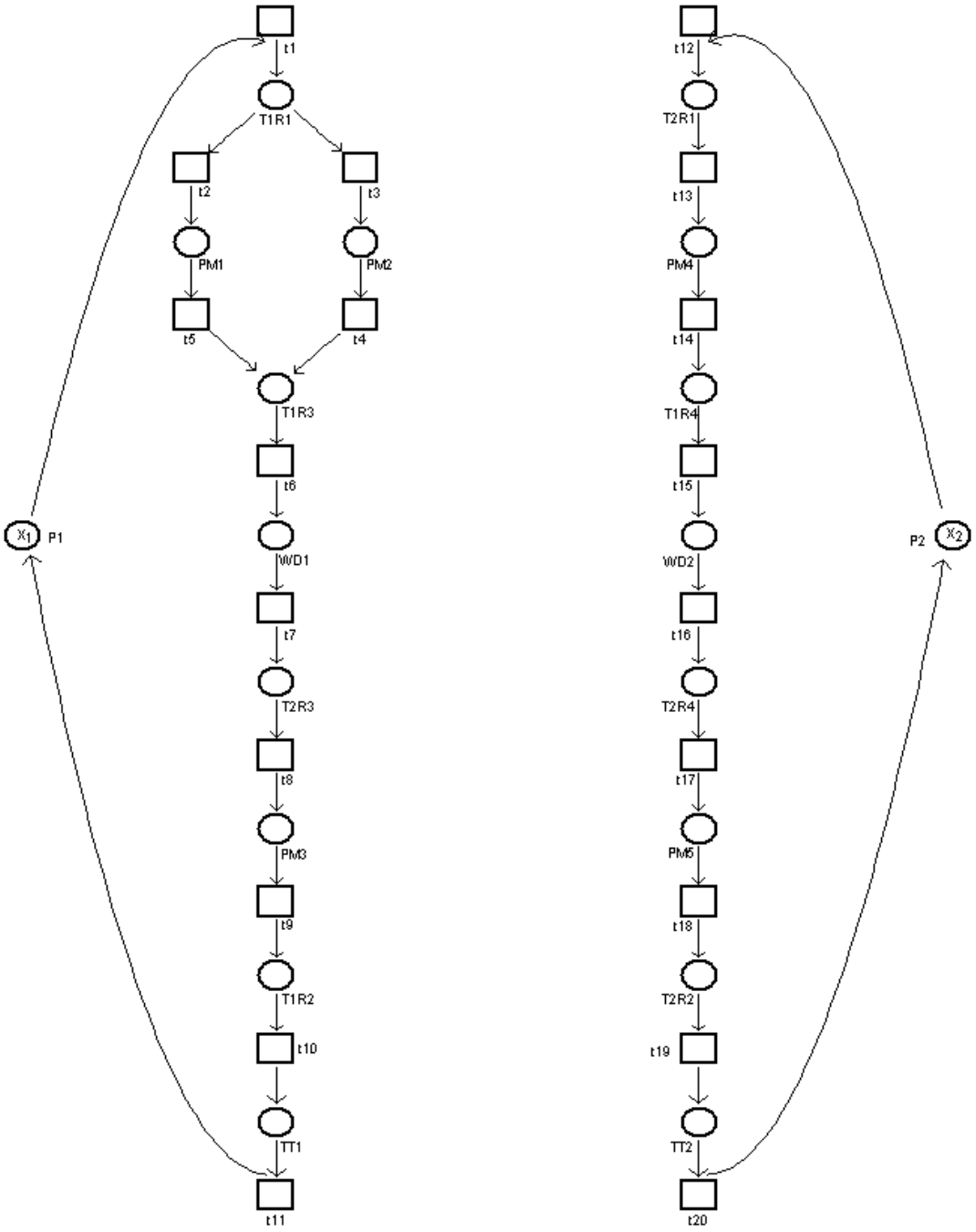


Fig. AP4.3.2. *Modelarea operațiilor în etapa de sinteză descendentă.*

Pasul 1: Ataşăm resursele specifice nepartajate:

- Pentru fluxul $PB_1 \rightarrow PF_1$: M_1, M_2, M_3 şi T_1, modelate prin pozițiile cu acelaşi nume;
- Pentru fluxul $PB_2 \rightarrow PF_2$: M_4, M_5, şi T_2, modelate prin pozițiile cu acelaşi nume.

Rețeaua obținută în urma aplicării *Pasului 1* al sintezei ascendente este prezentată în fig. AP4.3.3.

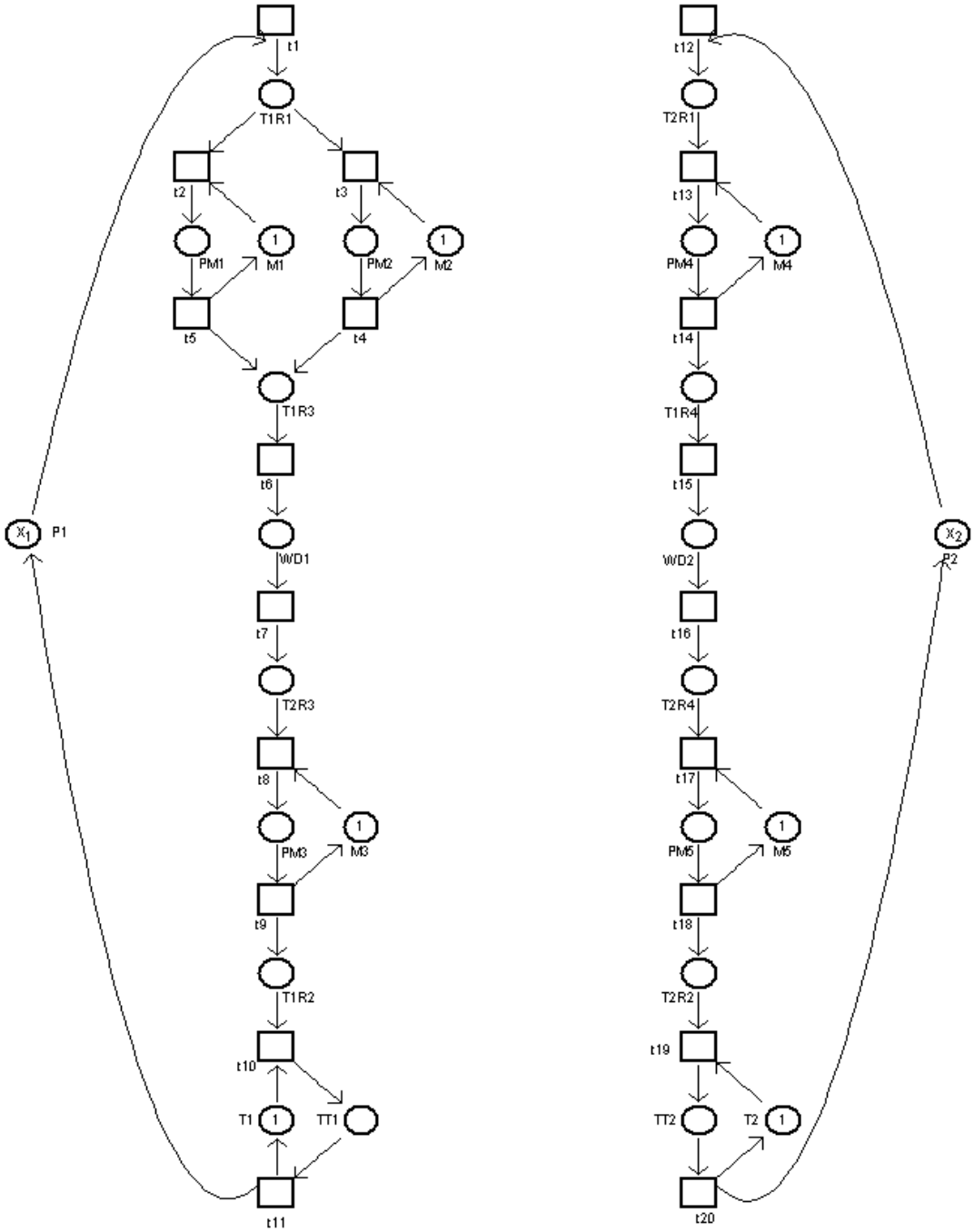


Fig. AP4.3.3. *Modelul rezultat după aplicarea Pasului 1 al etapei de sinteză ascendentă.*

Pasul 2: Ataşăm resursele de stocare.

Subpasul 2.1. Ataşăm resursele de stocare nepartajate D_1 (modelată prin D1) şi D_2 (modelată prin D2), rezultând reţeaua Petri din fig. AP4.3.4.

Nu există resurse de stocare partajate, deci nu vom executa *Subpasul 2.2*.

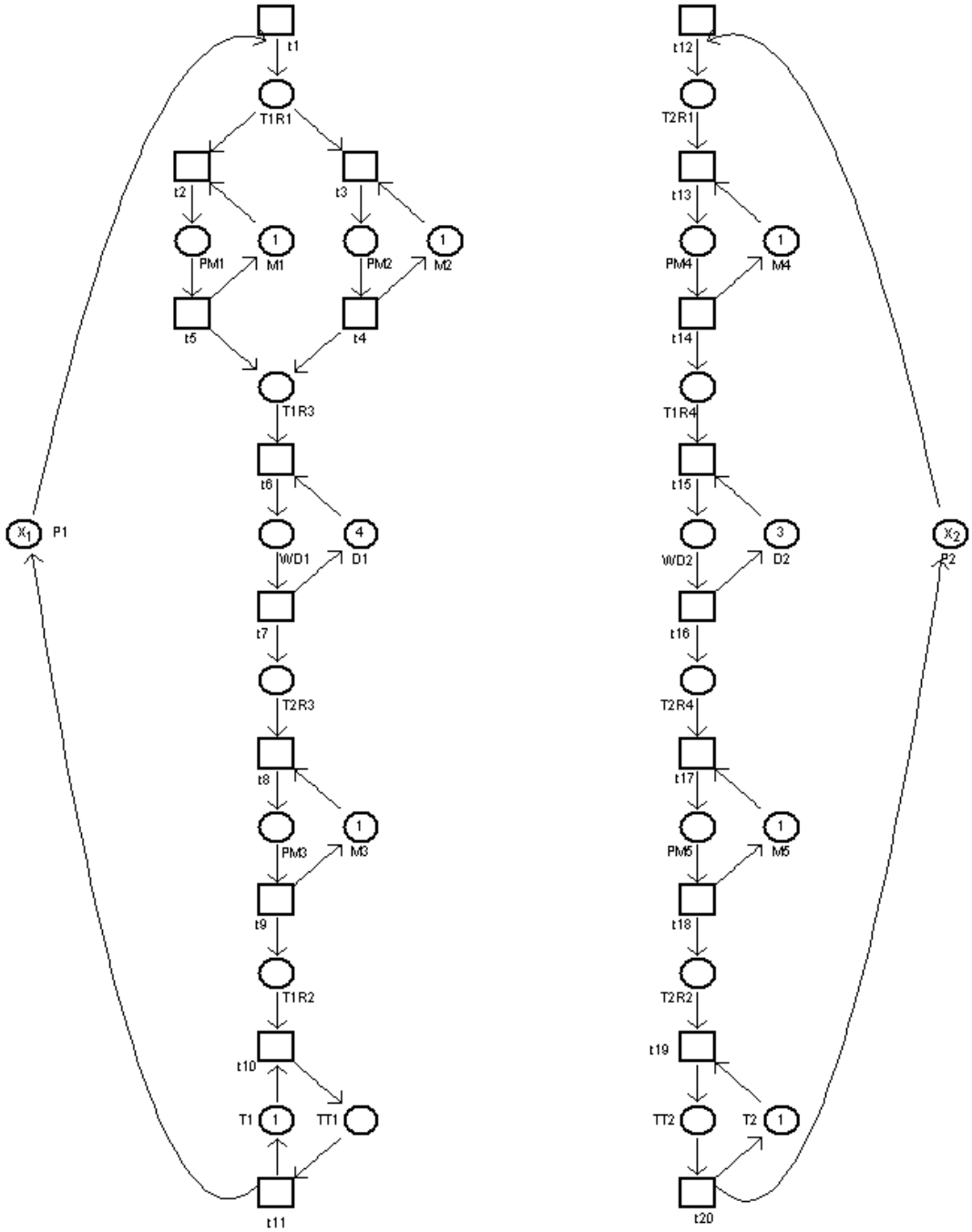


Fig. AP4.3.4. *Modelul rezultat după aplicarea Pasului 2 al etapei de sinteză ascendentă*.

Pasul 3. Atașăm resursele specifice partajate.

Subpasul 3.1. Atașăm resursele specifice partajate paralel, în maniera următoare:

- Poziția care modelează disponibilitatea robotului R_1, împreună cu perechile de tranziții $\{(t_1, t_2), (t_1, t_3), (t_{12}, t_{13})\}$, formează o excludere mutuală paralelă;
- Poziția care modelează disponibilitatea robotului R_2, împreună cu perechile de tranziții $\{(t_9, t_{10}), (t_{18}, t_{19})\}$, formează o excludere mutuală paralelă.

Rețeaua rezultată în urma aplicării S*ubpasului 3.1* este prezentată în fig. AP4.3.5 şi este viabilă, mărginită şi reversibilă.

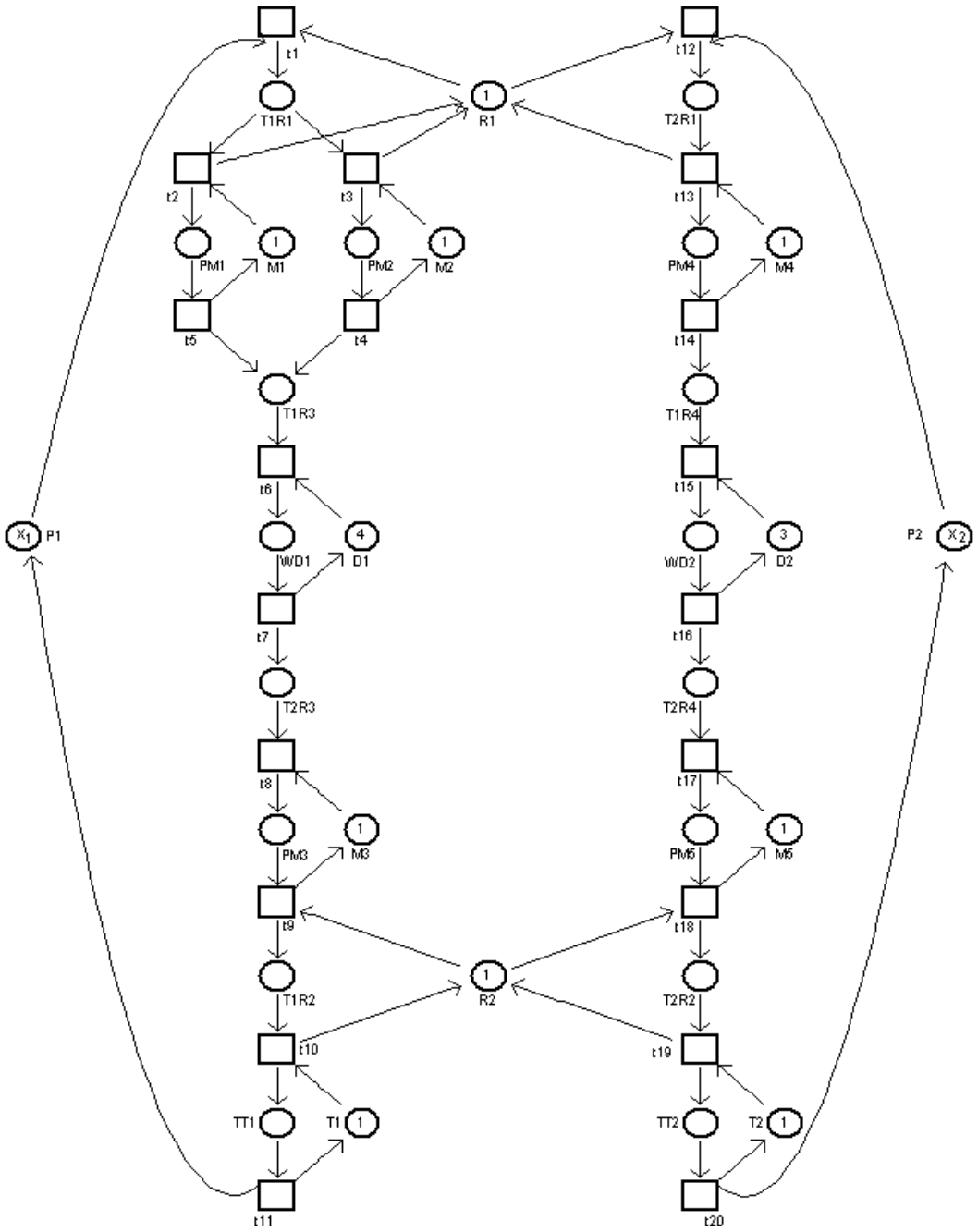


Fig. AP4.3.5. *Modelul rezultat după Subpasului 3.1 al etapei de sinteză ascendentă*.

Subpasul 3.2. Ataşăm robotul R_4 care este o resursă specifică partajată secvențial. Pentru ca poziția ce modelează disponibilitatea robotului R_4, împreună cu perechile de tranziții $\{(t_{14}, t_{15}), (t_{16}, t_{17})\}$, să formeze o excludere mutuală secvențială, aplicăm *Metoda 1* (de limitare a numărului total de clienți), care conduce la un număr maxim de palete $x_2 = 3$ pentru fluxul $PB_2 \rightarrow PF_2$.

Reţeaua rezultată în urma aplicării *Subpasului 3.2* este prezentată în fig. AP4.3.6. şi posedă proprietăţile de viabilitate, mărginire şi reversibilitate.

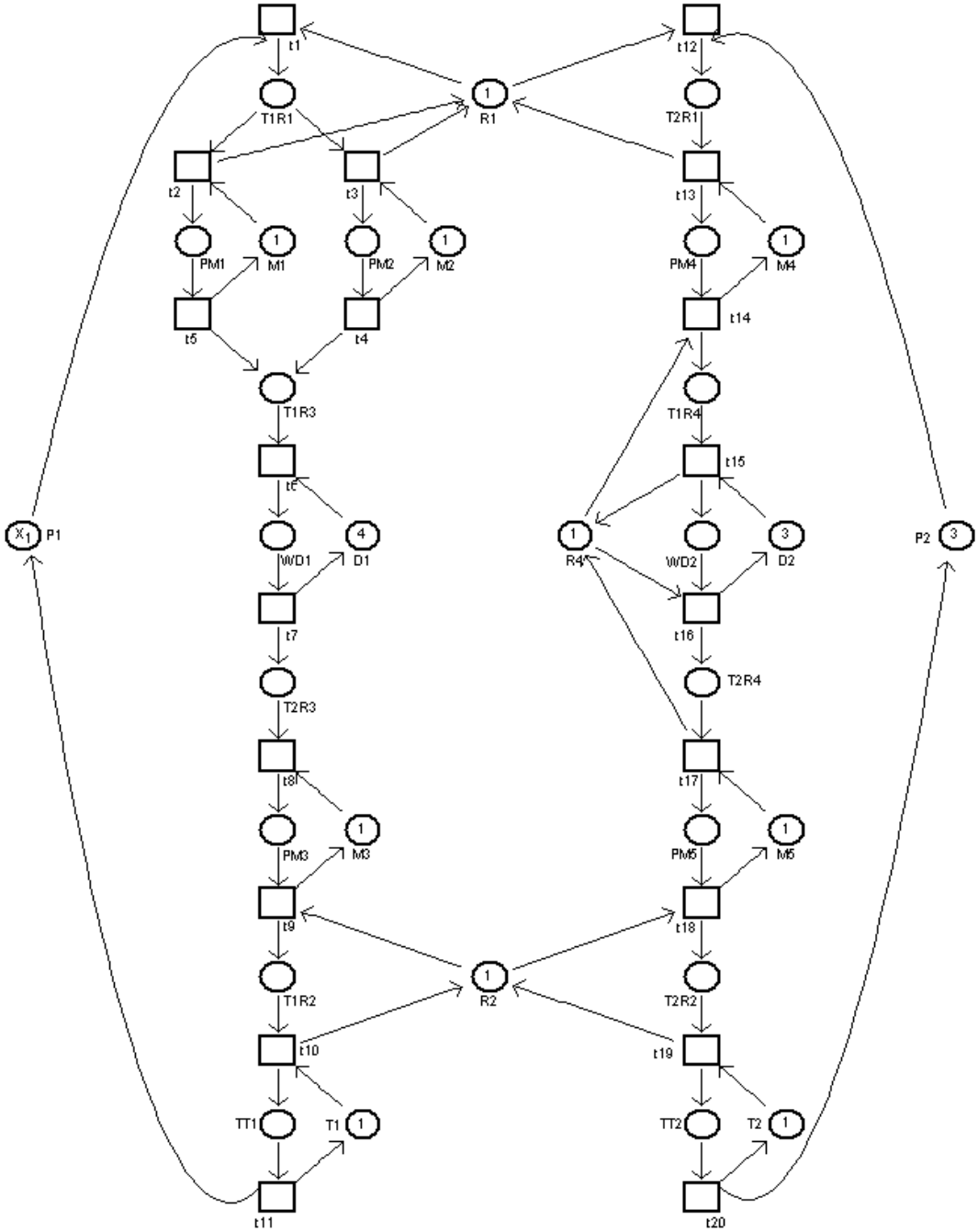


Fig. AP4.3.6. *Modelul rezultat după aplicarea Subpasului 3.2 al etapei de sinteză ascendentă.*

Subpasul 3.3. Ataşăm robotul R_3 care este o resursă specifică partajată generalizat. Trebuie ca poziţia R3 (ce modelează disponibilitatea robotului R_3), împreună cu perechile de tranziţii $\{(t_5, t_6), (t_4, t_6), (t_7, t_8)\}$, să formeze o excludere mutuală generalizată. Se constată că R3, împreună cu $\{(t_5, t_6), (t_4, t_6)\}$, formează o excludere mutuală paralelă. Mai trebuie ca R3, împreună cu $\{(t_5, t_6), (t_7, t_8)\}$, şi R3, împreună cu $\{(t_4, t_6), (t_7, t_8)\}$, să formeze excluderi

mutuale secvențiale. Utilizarea *Metodei 1* (de limitare a numărului total de clienți) conduce la un număr maxim de palete $x_1 = 4$ pentru fluxul $PB_1 \rightarrow PF_1$.

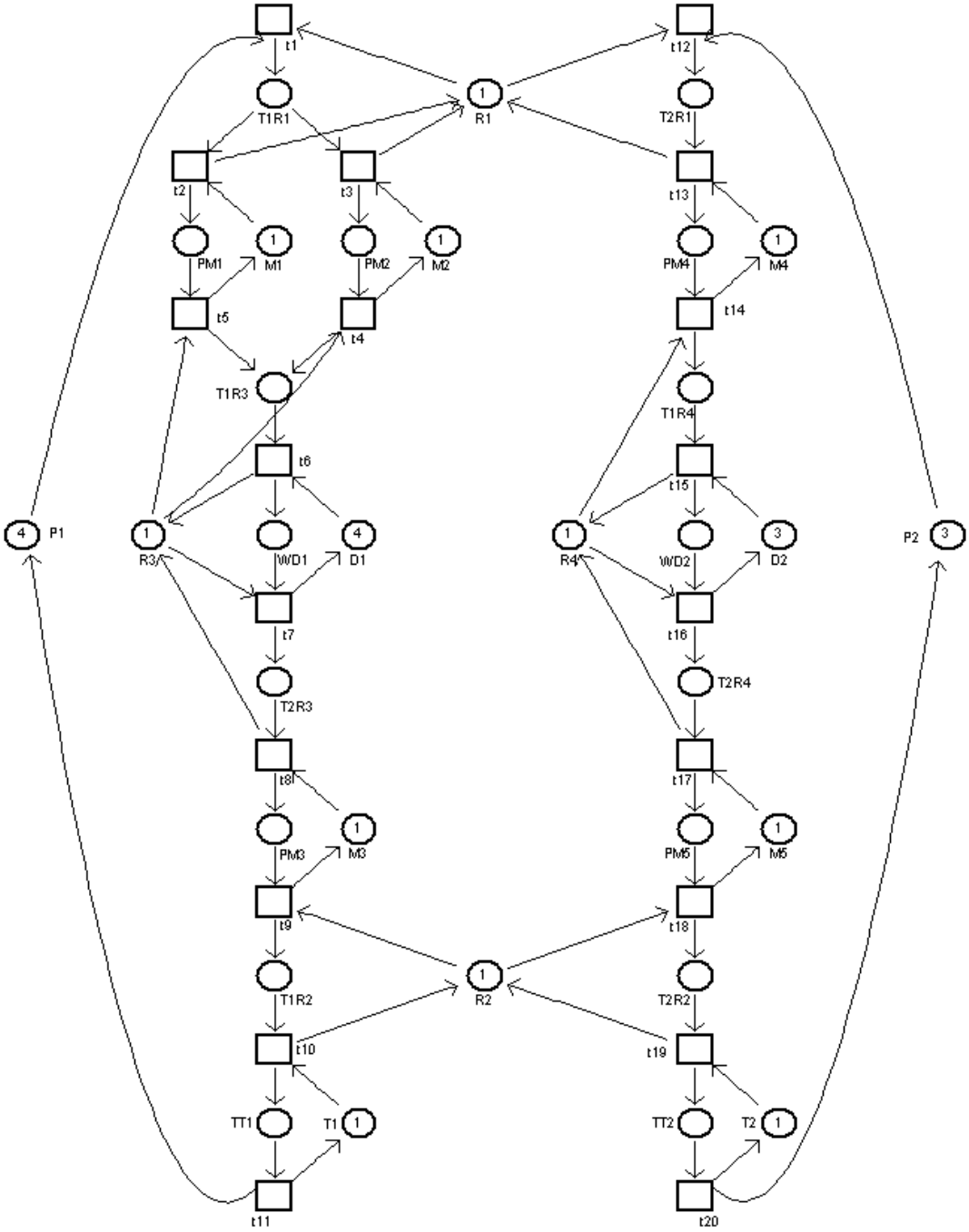


Fig. AP4.3.7. *Modelul rezultat după aplicarea Subpasului 3.3 al etapei de sinteză ascendentă, model ce reprezintă structura de conducere a sistemului de fabricație flexibil.*

Rețeaua rezultată în urma aplicării *Subpasului 3.3* este prezentată în fig. AP4.3.7 şi constituie structura de conducere a sistemului de fabricație flexibil. Rețeaua este viabilă, mărginită şi reversibilă.

2. Utilizând mediul ***Petri Net Toolbox*** putem verifica funcționarea corectă a structurii de comandă. În urma simulării unui număr de 10.000 de evenimente, se obțin valorile din fig. AP4.3.8.(a) pentru indicatorul ***Service Sum*** referitor la tranzițiile rețelei.

Petri Net Toolbox - Web Browser

Global S

Model:cap4pr3_pas4.xml

Events:10000

Time:0

Transition Name	Service Sum
t1	584
t2	282
t3	302
t4	301
t5	282
t6	583
t7	581
t8	581
t9	581
t10	581
t11	580
t12	531
t13	530
t14	530
t15	529
t16	529
t17	529
t18	528
t19	528
t20	528

(a)

Petri Net Toolbox - Web Browser

Global S

Model:cap4pr3_prio.xml

Events:10000

Time:0

Transition Name	Service Sum
t1	552
t2	286
t3	265
t4	265
t5	286
t6	551
t7	551
t8	550
t9	549
t10	549
t11	549
t12	562
t13	562
t14	561
t15	561
t16	561
t17	561
t18	560
t19	560
t20	559

(b)

Fig. AP4.3.8. *Indicatorul* ***Service Sum*** *pentru tranzițiile rețelei din fig. AP4.3.7*
(a) *în cazul* $x_1 = 4$, $x_2 = 3$ *;* (b) *în cazul* $x_1 = x_2 = 6$ (*cu priorități*).

Pe baza acestor valori rezultă:

i) numărul de piese transportate de R_1 = ***Service Sum***(t_2) + ***Service Sum***(t_3) + ***Service Sum***(t_{13}) = 282 + 302 + 530 = 1.114;

ii) numărul de piese prelucrate pe M_1 = ***Service Sum***(t_5) = 282;
iii) numărul de piese prelucrate pe M_2 = ***Service Sum***(t_4) = 301;
iv) numărul de piese prelucrate pe M_4 = ***Service Sum***(t_{14}) = 530.

Cititorul este invitat să formuleze o soluție bazată pe consultarea indicatorilor globali de performanță referitori la pozițiile din rețea.

3. Robotul R_1 este alocat prin executarea uneia dintre tranzițiile t_1 şi t_{12}.
Regula pentru executarea lui t_1:
IF (R_1 disponibil AND paletă disponibilă)
THEN (alocă robot R_1 AND încărcare pe M_1 sau M_2)
Regula pentru executarea lui t_{12}:
IF (R_1 disponibil AND paletă disponibilă)
THEN (alocă robot R_1 AND încărcare pe M_4)

Robotul R_1 este eliberat prin executarea uneia dintre tranzițiile t_2, t_3 şi t_{13}.
Regula pentru executarea lui t_2:
IF (M_1 disponibilă AND transport pe M_1 complet)
THEN (eliberează R_1 AND alocă M_1 AND începe prelucrare pe M_1)
Regula pentru executarea lui t_3:
IF (M_2 disponibilă AND transport pe M_2 complet)
THEN (eliberează R_1 AND alocă M_2 AND începe prelucrare pe M_2)
Regula pentru executarea lui t_{13}:
IF (M_4 disponibilă AND transport pe M_4 complet)
THEN (eliberează R_1 AND alocă M_4 AND începe prelucrare pe M_4)

4. Prin scăderea numărului de palete pe unul dintre fluxurile sistemului de fabricație, proprietățile de mărginire, viabilitate şi reversibilitate ale modelului nu sunt alterate. Rețelele corespunzătoare situațiilor de la punctele (i) şi (ii) sunt mărginite, viabile şi reversibile.

Prin creşterea numărului de palete pe unul dintre fluxurile sistemului, proprietatea de mărginire a rețelei Petri nu este alterată, în timp ce proprietățile de viabilitate şi reversibilitate ale modelului se pierd. Astfel, dacă folosim 5 palete pentru piese de tip PB_1 (iii) se poate observa că fluxul 1 se va bloca. Atunci când se folosesc 4 palete pentru piese de tip PB_2 (iv), fluxul 2 se va bloca. În cazul utilizării a 5 palete pentru piese de tip PB_1 şi 4 palete pentru piese de tip PB_2 (v) se va bloca întregul sistem.

5. Blocarea primului flux se produce prin alocarea necorespunzătoare a robotului R_3 (resursă partajată secvențial). Dacă impunem ca robotul R_3 să fie alocat cu prioritate pentru operația de încărcare a maşinii M_3 şi nu pentru descărcarea maşinii M_1 sau M_2, putem evita blocarea fluxului 1, indiferent de numărul de palete utilizate. Pe modelul de tip rețea Petri aceasta se realizează asignând prioritate de executare tranziției t_7 față de tranzițiile t_4 şi t_5.

Pentru cel de-al doilea flux blocarea circulară a resurselor se poate produce prin alocarea necorespunzătoare a robotului R_4 (resursă partajată secvențial). Considerând executarea tranziției t_{16} prioritară față de cea a tranziției t_{14}, blocarea circulară a resurselor este evitată, indiferent de numărul de palete utilizate.

6. După simularea apariției unui număr de 10.000 de evenimente în rețeaua Petri ce modelează funcționarea structurii de conducere ce utilizează câte 6 palete pe ambele fluxuri de fabricație, în cazul în care alocarea roboților R_3 şi R_4 se realizează pe baza strategiilor de priorități studiate la punctul 5, se obțin valorile prezentate în fig. AP4.3.7.(b). Pe baza acestora se pot obține:

i) numărul de piese transportate de R_1 = ***Service Sum***(t_2) + ***Service Sum***(t_3) + ***Service Sum***(t_{13}) = 286 + 265 + 562 = 1.113;

ii) numărul de piese prelucrate pe M_1 = ***Service Sum***(t_5) = 286;

iii) numărul de piese prelucrate pe M_2 = ***Service Sum***(t_4) = 265;

iv) numărul de piese prelucrate pe M_4 = ***Service Sum***(t_{14}) = 561.

Cititorul este invitat să formuleze o soluție bazată pe consultarea indicatorilor globali de performanță referitori la pozițiile din rețea.

Capitolul 5

Studierea proprietăților structurale

Breviar teoretic

BT5.1. Definirea proprietăților structurale şi criterii de caracterizare

Proprietățile *structurale* (eng. *structural*) ale rețelelor Petri netemporizate depind numai de topologia rețelei şi nu depind de *marcajul inițial* într-unul din sensurile următoare:

- fie proprietatea se păstrează indiferent de marcajul inițial;
- fie proprietatea se referă la faptul că există marcaje inițiale care asigură anumite secvențe de executări de tranziții.

Această dependență numai de topologie face ca tehnicile de analiză a proprietăților structurale să utilizeze *instrumente specifice algebrei liniare* care exploatează proprietățile *matricei de incidență* (vezi paragraful BT3.4).

În cele ce urmează, $\mathbb{Z}^m$ ($\mathbb{R}^m$) şi $\mathbb{Z}^{n\times m}$, pentru $n,m\in\mathbb{N}^*$, reprezintă mulțimea vectorilor coloană cu m elemente numere întregi (reale), respectiv cea a matricelor de dimensiune $n\times m$ cu elemente numere întregi. Vectorul cu toate elementele nule se notează cu $\boldsymbol{0}$. Fie $\boldsymbol{x},\boldsymbol{y}\in\mathbb{Z}^m$ doi vectori de aceeaşi dimensiune, $\boldsymbol{x}=[x_1,x_2,\ldots,x_m]^T$, $\boldsymbol{y}=[y_1,y_2,\ldots,y_m]^T$; în continuare vom utiliza următoarele notații *de tip inegalitate* între vectorii $\boldsymbol{x}$ şi $\boldsymbol{y}$:

$$\boldsymbol{x}<\boldsymbol{y} \quad\Leftrightarrow\quad x_i<y_i,\ \forall i=\overline{1,m};$$

$$\boldsymbol{x}\le\boldsymbol{y} \quad\Leftrightarrow\quad x_i\le y_i,\ \forall i=\overline{1,m};$$

$$\boldsymbol{x}<\!\!\neq\boldsymbol{y} \quad\Leftrightarrow\quad \boldsymbol{x}\le\boldsymbol{y} \ \text{ şi } \ \exists i\in\{1,2,\ldots,m\},\ x_i\neq y_i.$$

BT5.1.1. Mărginire structurală

Se spune că o *rețea Petri* $\boldsymbol{N}$ cu marcajul inițial M_0 este *structural mărginită* dacă este mărginită (în sens comportamental) pentru orice marcaj inițial finit.

Se spune că *o poziție p* a unei rețele Petri este *structural nemărginită* dacă există un marcaj M_0 şi o secvență de executări de tranziții σ pornind din M_0, astfel încât conținutul în jetoane al poziției *p* să devină nemărginit.

Teorema 5.1.1. O reţea Petri $\boldsymbol{N}$ este structural mărginită dacă şi numai dacă există un vector m–dimensional, $\boldsymbol{y} > \boldsymbol{0}$, cu toate elementele numere întregi, astfel încât:

$$\boldsymbol{A}\boldsymbol{y} \le \boldsymbol{0} . \qquad \text{(BT5.1.1)}$$

Teorema 5.1.2. Într-o reţea Petri $\boldsymbol{N}$, o poziţie p este structural nemărginită dacă şi numai dacă există un vector n–dimensional $\boldsymbol{x} \ge \boldsymbol{0}$, cu toate elementele numere întregi, astfel încât:

$$\boldsymbol{A}^T\boldsymbol{x} = \Delta M >\neq \boldsymbol{0} \quad \text{şi} \quad \Delta M(p) > \boldsymbol{0} . \qquad \text{(BT5.1.2)}$$

Observaţie: Relaţia (BT5.1.2) are semnificaţia următoare: elementul vectorului $\boldsymbol{A}^T\boldsymbol{x}$, corespunzător poziţiei p, este pozitiv, celelalte elemente putând fi nule.

BT5.1.2. Conservativitate

Se spune că o reţea Petri $\boldsymbol{N}$ este *conservativă* dacă există câte un întreg pozitiv $\boldsymbol{y}(p)$ pentru fiecare poziţie p, astfel încât pentru orice marcaj iniţial M_0 fixat şi orice marcaj accesibil din M_0, $M \in R(M_0)$, are loc egalitatea:

$$M^T\boldsymbol{y} = M_0^T\boldsymbol{y} (= \text{constant}) . \qquad \text{(BT5.1.3)}$$

Se spune că o reţea Petri $\boldsymbol{N}$ este *parţial conservativă* dacă există câte un întreg pozitiv $\boldsymbol{y}(p)$ pentru o parte din poziţiile p (în sensul că nu pentru toate poziţiile), astfel încât pentru orice marcaj iniţial M_0 fixat şi orice marcaj $M \in R(M_0)$, are loc egalitatea (BT5.1.3).

Observaţie: Egalitatea (BT5.1.3) are semnificaţia unei sume ponderate de jetoane care se păstrează pentru orice marcaj accesibil. Atragem atenţia că deşi egalitatea (BT5.1.3) este una şi aceeaşi pentru conservativitate şi conservativitate parţială, în cazul conservativităţii relaţia (BT5.1.3) cere $\boldsymbol{y} > \boldsymbol{0}$ (vectorial), iar în cazul conservativităţii parţiale, relaţia (BT5.1.3) cere numai $\boldsymbol{y} >\neq \boldsymbol{0}$ (vectorial).

Teorema 5.1.3. O reţea Petri $\boldsymbol{N}$ este conservativă dacă şi numai dacă există un vector m–dimensional $\boldsymbol{y} > \boldsymbol{0}$, cu elementele numere întregi, astfel încât:

$$\boldsymbol{A}\boldsymbol{y} = \boldsymbol{0} . \qquad \text{(BT5.1.4)}$$

Teorema 5.1.4. O reţea Petri $\boldsymbol{N}$ este parţial conservativă dacă şi numai dacă există un vector m–dimensional $\boldsymbol{y} >\neq \boldsymbol{0}$, cu elementele numere întregi, astfel încât să fie satisfăcută egalitatea (BT5.1.4).

BT5.1.3. Repetitivitate

Se spune că o reţea Petri $\boldsymbol{N}$ este *repetitivă* dacă există un marcaj iniţial M_0 şi o secvenţă de executări de tranziţii σ ce porneşte din M_0, astfel încât orice tranziţie apare infinit de des în σ.

Se spune că o reţea Petri $\boldsymbol{N}$ este *parţial repetitivă* dacă există un marcaj iniţial M_0 şi o secvenţă de executări de tranziţii σ ce porneşte din M_0, astfel încât o parte din tranziţii (în sensul că nu toate tranziţiile) apar infinit de des în σ.

Teorema 5.1.5. O rețea Petri ***N*** este repetitivă dacă şi numai dacă există un vector *n*–dimensional $\boldsymbol{x} > \boldsymbol{0}$, cu elemente numere întregi, astfel încât:

$$\boldsymbol{A}^T \boldsymbol{x} \geq \boldsymbol{0}. \qquad \text{(BT5.1.5)}$$

Teorema 5.1.6. O rețea Petri ***N*** este parțial repetitivă dacă şi numai dacă există un vector *n*–dimensional $\boldsymbol{x} >\neq \boldsymbol{0}$, cu elementele numere întregi, astfel încât condiția (BT5.1.5) să fie îndeplinită.

BT5.1.4. Consistență

Se spune că o rețea Petri este *consistentă* dacă există un marcaj M_0 şi o secvență de executări de tranziții σ de la M_0 înapoi la M_0, astfel încât fiecare tranziție apare cel puțin o dată în σ.

Se spune că o rețea Petri este *parțial consistentă* dacă există un marcaj M_0 şi o secvență de executări de tranziții σ de la M_0 înapoi la M_0, astfel încât o parte din tranziții (în sensul că nu toate) apar cel puțin o dată în σ.

Teorema 5.1.7. O rețea Petri ***N*** este consistentă dacă şi numai dacă există un vector *n*–dimensional $\boldsymbol{x} > \boldsymbol{0}$, cu elementele numere întregi, astfel încât:

$$\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}. \qquad \text{(BT5.1.6)}$$

Teorema 5.1.8. O rețea Petri ***N*** este parțial consistentă dacă şi numai dacă există un vector *n*–dimensional $\boldsymbol{x} >\neq \boldsymbol{0}$, cu toate elementele întregi, astfel încât să fie îndeplinită condiția (BT5.1.6).

BT5.1.5. Relații între proprietățile structurale

Tabelul BT5.1.1 prezintă rezumativ condițiile necesare şi suficiente pentru existența proprietăților structurale prezentate anterior. În aceste inegalități, vectorul $\boldsymbol{A}^T \boldsymbol{x}$ trebuie privit ca diferență de marcaj corespunzând celor *m* poziții, iar vectorul $\boldsymbol{Ay}$ trebuie interpretat ca schimbare într-o sumă ponderată de jetoane corespunzând executării celor *n* tranziții ale rețelei Petri.

Tab. BT5.1.1. *Prezentare rezumativă a criteriilor de tip necesar şi suficient pentru existența unor proprietăți structurale.*

Proprietate structurală	Condiție necesară şi suficientă	Teorema
N structural mărginită	$\exists\, \boldsymbol{y} > \boldsymbol{0}$ a.î. $\boldsymbol{Ay} \leq \boldsymbol{0}$	**5.1.1**
N conservativă	$\exists\, \boldsymbol{y} > \boldsymbol{0}$ a.î. $\boldsymbol{Ay} = \boldsymbol{0}$	**5.1.3**
N parțial conservativă	$\exists\, \boldsymbol{y} >\neq \boldsymbol{0}$ a.î. $\boldsymbol{Ay} = \boldsymbol{0}$	**5.1.4**
N repetitivă	$\exists\, \boldsymbol{x} > \boldsymbol{0}$ a.î. $\boldsymbol{A}^T \boldsymbol{x} \geq \boldsymbol{0}$	**5.1.5**
N parțial repetitivă	$\exists\, \boldsymbol{x} >\neq \boldsymbol{0}$ a.î. $\boldsymbol{A}^T \boldsymbol{x} \geq \boldsymbol{0}$	**5.1.6**
N consistentă	$\exists\, \boldsymbol{x} > \boldsymbol{0}$ a.î. $\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}$	**5.1.7**
N parțial consistentă	$\exists\, \boldsymbol{x} >\neq \boldsymbol{0}$ a.î. $\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}$	**5.1.8**

În baza definiţiilor proprietăţilor structurale din secţiunile anterioare se constată că unele din aceste proprietăţi constituie *cazuri particulare* ale altora. Cu alte cuvinte *dacă* o reţea Petri ***N*** posedă o anumită proprietate structurală, *atunci* va poseda şi o altă proprietate structurală, conform implicaţiilor prezentate rezumativ în tabelul BT5.1.2.

Tab. BT5.1.2. *Prezentare rezumativă a conexiunilor de tip* ***dacă–atunci*** *existente între proprietăţile structurale.*

Implicaţie	**Dacă** ⇒	**Atunci**
1.	***N*** conservativă	***N*** structural mărginită
2.	***N*** consistentă	***N*** repetitivă
3.	***N*** conservativă	***N*** parţial conservativă
4.	***N*** repetitivă	***N*** parţial repetitivă
5.	***N*** consistentă	***N*** parţial consistentă
6.	***N*** parţial consistentă	***N*** parţial repetitivă

Referitor la ilustrarea proprietăţilor structurale se vor consulta **AP5.1** – **AP5.4**.

BT5.2. Invarianţi

În secţiunea de faţă continuăm *explorarea algebrică* a proprietăţilor structurale, în scopul identificării acelor *elemente topologice* (submulţimi de poziţii şi submulţimi de tranziţii) ce asigură conservativitatea (eventual parţială) şi respectiv consistenţa (eventual parţială) a unei reţele Petri.

BT5.2.1. Definiţia invarianţilor

Fie o reţea Petri ***N***, a cărei topologie este descrisă prin matricea de incidenţă $\boldsymbol{A} \in \mathbb{Z}^{n \times m}$. Un vector *m*–dimensional $\boldsymbol{y} \geqslant \neq \boldsymbol{0}$, cu elementele numere întregi se numeşte *invariant P* al reţelei dacă $\boldsymbol{Ay} = \boldsymbol{0}$. Un vector *n*–dimensional $\mathbf{x} \geqslant \neq \mathbf{0}$, cu elementele numere întregi, se numeşte *invariant T* al reţelei dacă $\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}$. Este evident faptul că pentru o reţea Petri pot exista mai mulţi invarianţi *P* şi/sau mai mulţi invarianţi *T*.

Mulţimea poziţiilor reţelei ***N*** care corespund elementelor nenule ale unui invariant *P*, notat ***y***, se numeşte *suportul invariantului* ***y*** şi se notează cu <***y***>. *Mulţimea tranziţiilor* reţelei ***N*** care corespund elementelor nenule ale unui invariant *T*, notat ***x***, se numeşte *suportul invariantului* ***x*** şi se notează cu <***x***>.

Un suport de invariant (adică o mulţime de poziţii, respectiv de tranziţii) se numeşte *minimal* dacă nici o *submulţime strictă* a suportului nu reprezintă tot un suport (evident pentru un alt invariant).

Un invariant *P*, notat $\boldsymbol{y}$, se numeşte *invariant P minimal*, dacă nu există nici un alt invariant *P*, notat $\boldsymbol{y}'$, astfel încât $\boldsymbol{y} >\neq \boldsymbol{y}'$. Un invariant *T*, notat $\boldsymbol{x}$, se numeşte *invariant T minimal*, dacă nu există nici un alt invariant *T*, notat $\boldsymbol{x}'$, astfel încât $\boldsymbol{x} >\neq \boldsymbol{x}'$.

Nu orice *invariant* (*P sau T*) *minimal* are suportul minimal (adică pot exista invarianți (*P* sau *T*) minimali având suporturi neminimale). Nu orice *invariant* (*P sau T*) *cu suport minimal* este minimal (adică pot exista invarianți (*P* sau *T*) cu suport minimal care nu sunt invarianți minimali).

Un invariant minimal cu suport minimal se numeşte *invariant de bază* (*fundamental*). Aşadar fiecărei mulțimi (de poziții sau tranziții) care constituie un suport minimal îi corespunde un singur invariant (*P* sau *T*) de bază (fundamental) şi reciproc.

Teorema 5.2.1. Dacă matricea de incidență a rețelei $\boldsymbol{A} \in \mathbb{Z}^{n\times m}$ are rangul *r*, atunci:
1. Rețeaua posedă cel mult *m–r* invarianți *P* liniar independenți.
2. Rețeaua posedă cel mult *n–r* invarianți *T* liniar independenți.

Teorema 5.2.2. Fie $\boldsymbol{a}$, $\boldsymbol{b}$ doi invarianți de acelaşi tip (*P* sau *T*) ai unei rețele Petri $\boldsymbol{N}$. Dacă pentru $\alpha, \beta \in \mathbb{R}_+$, vectorul $\alpha\boldsymbol{a} + \beta\boldsymbol{b}$ are toate elementele întregi, nenegative, atunci:
1. $\alpha\boldsymbol{a} + \beta\boldsymbol{b}$ este un invariant (*P* respectiv *T*) al lui $\boldsymbol{N}$.
2. $<\alpha\boldsymbol{a} + \beta\boldsymbol{b}> = <\boldsymbol{a}> \cup <\boldsymbol{b}>$.

BT5.2.2. Invarianți în rețele duale şi/sau inversate

Fie o rețea Petri $\boldsymbol{N}$ şi o rețea $\boldsymbol{N}_{di}$, construită cu ajutorul lui $\boldsymbol{N}$, astfel încât:
(*i*) Nodurile de tip poziție ale lui $\boldsymbol{N}_{di}$ sunt nodurile de tip tranziție ale lui $\boldsymbol{N}$.
(*ii*) Nodurile de tip tranziție ale lui $\boldsymbol{N}_{di}$ sunt nodurile de tip poziție ale lui $\boldsymbol{N}$.
(*iii*) Arcele lui $\boldsymbol{N}_{di}$ sunt arcele lui $\boldsymbol{N}$, dar având sensul inversat.

Rețeaua $\boldsymbol{N}_{di}$ se numeşte *rețeaua dual inversată* a lui $\boldsymbol{N}$. Atragem atenția asupra semnificației acestei construcții şi a terminologiei aferente:

- Pornind de la rețeaua $\boldsymbol{N}$ şi aplicând (*i*), (*ii*) (fără a inversa sensul arcelor, adică fără (*iii*)), se obține *rețeaua dual* $\boldsymbol{N}_d$.
- Pornind de la rețeaua $\boldsymbol{N}$ şi schimbând sensul arcelor (adică fără (*i*), (*ii*)), se obține *rețeaua inversată* $\boldsymbol{N}_i$.

Conexiunea dintre topologiile rețelelor $\boldsymbol{N}_d$, $\boldsymbol{N}_i$, $\boldsymbol{N}_{di}$, şi a rețelei originale $\boldsymbol{N}$ este reflectată de relațiile dintre matricele lor de incidență, $\boldsymbol{A}(\boldsymbol{N}_d)$, $\boldsymbol{A}(\boldsymbol{N}_i)$, $\boldsymbol{A}(\boldsymbol{N}_{di})$, respectiv $\boldsymbol{A}(\boldsymbol{N})$:

$$\boldsymbol{A}(\boldsymbol{N}_d) = -[\boldsymbol{A}(\boldsymbol{N})]^T, \qquad \boldsymbol{A}(\boldsymbol{N}_i) = -\boldsymbol{A}(\boldsymbol{N}), \qquad \boldsymbol{A}(\boldsymbol{N}_{di}) = [\boldsymbol{A}(\boldsymbol{N})]^T.$$

În baza acestor relații, rezultă că problema invarianților are un caracter dual dacă operăm cu o rețea $\boldsymbol{N}$ şi duala inversată a acesteia $\boldsymbol{N}_{di}$ (ilustrare în **AP5.1**).

Teorema 5.2.3. Fie $\boldsymbol{N}$ o rețea Petri şi $\boldsymbol{N}_{di}$ duala inversată a acesteia. Atunci sunt adevărate următoarele afirmații:
1. Invarianții *P* ai lui $\boldsymbol{N}_{di}$ coincid cu invarianții *T* ai lui $\boldsymbol{N}$.
2. Invarianții *T* ai lui $\boldsymbol{N}_{di}$ coincid cu invarianții *P* ai lui $\boldsymbol{N}$.

BT5.2.3. Criterii de caracterizare a invarianţilor şi conexiuni cu proprietăţile structurale

Definiţiile introduse în paragraful BT5.2.1 pentru invarianţii *P* şi *T* fac posibilă formularea unor condiţii de tip necesar şi suficient care exploatează descrierea algebrică a topologiei reţelei. Aceste criterii evidenţiază legătura directă dintre *invarianţii P* şi *conservativitate*, pe de o parte, precum şi dintre *invarianţii T* şi *consistenţă*, pe de altă parte.

Teorema 5.2.4. Un vector *m*–dimensional $\boldsymbol{y} >\neq \boldsymbol{0}$, cu toate elementele întregi, este un invariant *P*, dacă şi numai dacă, pentru un marcaj iniţial M_0 arbitrar şi pentru orice marcaj accesibil din M_0, $M \in R(M_0)$, are loc egalitatea:

$$M^T \boldsymbol{y} = M_0^T \boldsymbol{y} (= \text{constant}) . \qquad \text{(BT5.2.1)}$$

Din relaţia (BT5.2.1) rezultă următoarea semnificaţie fizică a invarianţilor *P* de bază în modelarea dinamicii operaţiilor plus resurselor: fiecărei resurse din sistem i se poate asocia un singur invariant *P* de bază, iar egalitatea de mai sus are semnificaţia *conservării* acelei resurse pe parcursul funcţionării sistemului.

Teorema 5.2.5. Un vector *n*–dimensional $\boldsymbol{x} >\neq \boldsymbol{0}$, cu toate elementele întregi, este un invariant *T*, dacă şi numai dacă există un marcaj iniţial M_0 şi o secvenţă de executări σ, ce porneşte din M_0 şi ajunge înapoi la M_0, cu vectorul numărului de executări $\overline{\sigma} = \boldsymbol{x}$.

<u>*Observaţie:*</u> Din cele două teoreme de mai sus rezultă următoarele aspecte topologice fundamentale legate de existenţa invarianţilor:

- invarianţi *P* posedă numai reţelele Petri care sunt parţial conservative;
- invarianţi *T* posedă numai reţelele Petri care sunt parţial consistente.

Utilizând criteriul de caracterizare a invarianţilor *P* dat de **Teorema 5.2.4**, se poate determina *numărul maxim de jetoane* ce pot exista într-o poziţie *p*, aparţinând unuia sau mai multor suporturi minimale de invarianţi *P*.

Teorema 5.2.6. Fie p^* o poziţie oarecare a unei reţele Petri cu marcajul iniţial M_0. Fie $\boldsymbol{y}^i$, $i = 1,...,q$, cu $\boldsymbol{y}^i(p^*) > 0$, toţi invarianţii *P* de bază ai căror suporturi conţin poziţia p^*. Marcajul poziţiei p^* satisface inegalitatea:

$$M(p^*) \le \min_{i=1,...,q} \left[M_0^T \boldsymbol{y}^i \,/\, \boldsymbol{y}^i(p^*) \right]. \qquad \text{(BT5.2.2)}$$

Acest rezultat generalizează **Teorema 3.5.9** din cazul grafurilor marcate; într-un graf marcat poziţiile oricărui circuit elementar reprezintă un suport minimal de invariant *P*. Mai mult, topologia grafului marcat face ca elementele nenule ale oricărui invariant *P* de bază să fie egale cu 1. Din aceste considerente rezultă imediat că, pentru grafuri marcate, inegalitatea (BT5.2.2) se reduce la inegalitatea (BT3.5.3).

Se spune că o reţea Petri ***N*** este *acoperită de invarianţi P*, dacă *fiecare* poziţie a lui ***N*** aparţine suportului unui invariant *P* de bază. Se spune că o reţea Petri ***N*** este *acoperită de invarianţi T*, dacă *fiecare* tranziţie a lui ***N*** aparţine suportului unui invariant *T* de bază.

Rețeaua Petri ***N*** este acoperită de invarianți $P(T)$ dacă şi numai dacă ***N*** posedă invarianți $P(T)$ cu toate elementele nenule.

Teorema 5.2.7. Dacă ***N*** este o rețea Petri acoperită de invarianți P, atunci ***N*** este conservativă şi structural mărginită.

Teorema 5.2.8. Dacă ***N*** este o rețea Petri acoperită de invarianți T, atunci ***N*** este consistentă şi repetitivă.

Referitor la ilustrarea invarianților şi legătura lor cu proprietățile structurale se vor consulta **AP5.1** – **AP5.4**.

Aplicații

AP5.1.

Se consideră rețelele Petri cu topologiile din fig. AP5.1.1.

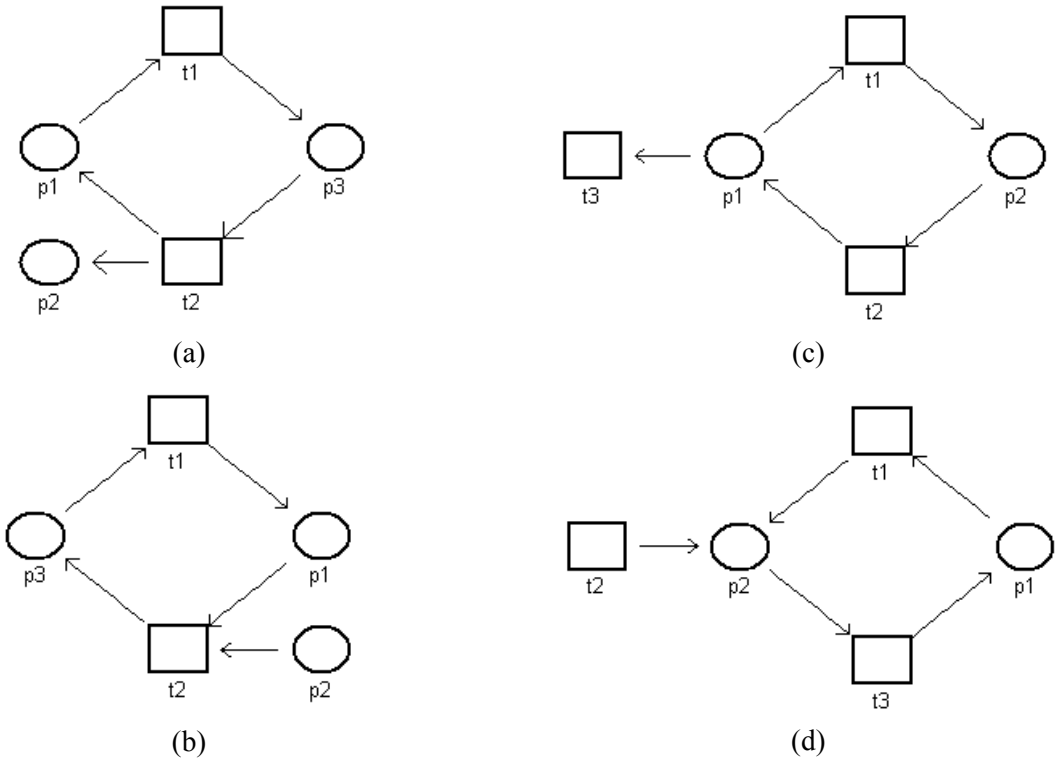


Fig. AP5.1.1. *Topologiile rețelelor Petri studiate în AP5.1.*

Pentru fiecare dintre aceste rețele să se abordeze următoarea problematică:

1. Să se determine matricea de incidență.

2. Aplicând criteriile algebrice, să se investigheze următoarele proprietăți structurale:
   (i) mărginire structurală;
   (ii) conservativitate;
   (iii) repetitivitate;
   (iv) consistență.
3. Să se determine invarianții *P* și *T* și să se pună în evidență legătura cu rezultatele obținute la punctul 2.

Soluție

A. Pentru rețeaua ***N*** din fig. AP5.1.1.(a) se obțin următoarele rezultate:
1. Matricele de incidență sunt date de:

$$\boldsymbol{A}^{+}=\begin{bmatrix}0 & 0 & 1\\ 1 & 1 & 0\end{bmatrix},\quad \boldsymbol{A}^{-}=\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix},\quad \boldsymbol{A}=\boldsymbol{A}^{+}-\boldsymbol{A}^{-}=\begin{bmatrix}-1 & 0 & 1\\ 1 & 1 & -1\end{bmatrix}. \qquad \text{(AP5.1.1)}$$

2.(i). Studierea proprietății de *mărginire structurală* revine la studierea compatibilității sistemului de inecuații liniare $\boldsymbol{A}\boldsymbol{y}\le\boldsymbol{0}$ cu $\boldsymbol{y}\in\mathbb{Z}^3$, $\boldsymbol{y}>\boldsymbol{0}$. Din relațiile

$$\boldsymbol{A}\boldsymbol{y}\le\boldsymbol{0}\Leftrightarrow\begin{cases}-y_1+y_3\le 0\\ y_1+y_2-y_3\le 0\end{cases}\Leftrightarrow\begin{cases}y_2\le 0\\ y_3\le y_1\end{cases} \qquad \text{(AP5.1.2)}$$

rezultă $y_2\le 0$, deci rețeaua ***N*** *nu este structural mărginită*. Mai exact, faptul că orice vector de forma $\boldsymbol{y}=\begin{bmatrix}\alpha & 0 & \beta\end{bmatrix}^T$, cu $\alpha,\beta\in\mathbb{Z}$, $\alpha\ge\beta>0$, satisface sistemul de inecuații arată că poziția p_2 este *structural nemărginită.* Într-adevăr, pentru orice marcaj inițial care plasează cel puțin un jeton în p_1 sau în p_3, pentru secvența în care se execută succesiv tranzițiile t_1 și t_2, conținutul în jetoane al poziției p_2 devine nemărginit.

(ii). Studierea proprietății de *conservativitate* revine la studierea compatibilității sistemului de ecuații liniare $\boldsymbol{A}\boldsymbol{y}=\boldsymbol{0}$, cu $\boldsymbol{y}\in\mathbb{Z}^3$, $\boldsymbol{y}>\boldsymbol{0}$. Deoarece

$$\boldsymbol{A}\boldsymbol{y}=\boldsymbol{0}\Leftrightarrow\begin{cases}y_2=0\\ y_1=y_3\end{cases} \qquad \text{(AP5.1.3)}$$

orice vector de forma $\boldsymbol{y}=\begin{bmatrix}\alpha & 0 & \alpha\end{bmatrix}^T$, cu $\alpha\in\mathbb{Z}$, $\alpha>0$, satisface sistemul investigat. Pentru că $y_2=0$, rețeaua ***N*** *nu este conservativă*, dar *este parțial conservativă*.

(iii). Studierea proprietății de *repetitivitate* revine la studierea compatibilității sistemului de inecuații liniare $\boldsymbol{A}^T\boldsymbol{x}\ge\boldsymbol{0}$, cu $\boldsymbol{x}\in\mathbb{Z}^2$, $\boldsymbol{x}>\boldsymbol{0}$:

$$\boldsymbol{A}^T\boldsymbol{x}\ge\boldsymbol{0}\Leftrightarrow\begin{cases}-x_1+x_2\ge 0\\ x_2\ge 0\\ x_1-x_2\ge 0\end{cases}\Leftrightarrow\begin{cases}x_1=x_2\\ x_2\ge 0\end{cases} \qquad \text{(AP5.1.4)}$$

Orice vector de forma $\boldsymbol{x}=\begin{bmatrix}\alpha & \alpha\end{bmatrix}^T$, cu $\alpha\in\mathbb{Z}$, $\alpha>0$, satisface condițiile impuse, deci rețeaua ***N*** *este structural repetitivă*.

(iv). Studierea proprietății de *consistență* revine la studierea compatibilității sistemului de ecuații liniare $\boldsymbol{A}^T\boldsymbol{x}=\boldsymbol{0}$, cu $\boldsymbol{x}\in\mathbb{Z}^2$, $\boldsymbol{x}>\boldsymbol{0}$. Din relația

$$\boldsymbol{A}^T\boldsymbol{x}=\boldsymbol{0} \Leftrightarrow \begin{cases} -x_1+x_2=0 \\ x_2=0 \\ x_1-x_2=0 \end{cases} \Leftrightarrow x_1=x_2=0, \qquad \text{(AP5.1.5)}$$

rezultă că rețeaua $\boldsymbol{N}$ *nu este* nici *consistentă*, nici *parțial consistentă*.

3. Din discuția de la punctul 2.(ii) rezultă că orice vector de forma $\boldsymbol{y}=\begin{bmatrix}\alpha & 0 & \alpha\end{bmatrix}^T$, cu $\alpha\in\mathbb{Z}$, $\alpha>0$, este un invariant de tip *P* al rețelei $\boldsymbol{N}$. Se observă că poziția p_2 nu aparține nici unui suport de invariant *P*. Din relația (AP5.1.5) de la punctul 2.(iv) rezultă că nu se poate pune în evidență nici un invariant *T* pentru rețeaua $\boldsymbol{N}$.

Într-o manieră similară se pot trata punctele 1 – 3 şi în cazul rețelelor din fig. AP5.1.1.(b) – (d). Vom lăsa acest tip de abordare drept exercițiu pentru cititor şi vom face apel la noțiunile de dualitate şi inversare privind topologiile rețelelor Petri (BT5.2.2).

B. Se observă că rețeaua Petri $\boldsymbol{N}_i$ din fig. AP5.1.1.(b) este *inversa* rețelei $\boldsymbol{N}$ din fig. AP5.1.1.(a), astfel că se obțin următoarele rezultate:
1. Matricele de incidență corespunzătoare lui $\boldsymbol{N}_i$ sunt date de:

$$\boldsymbol{A}_i^+=\boldsymbol{A}^-,\quad \boldsymbol{A}_i^-=\boldsymbol{A}^+ \;\Rightarrow\; \boldsymbol{A}_i=-\boldsymbol{A}. \qquad \text{(AP5.1.6)}$$

2.(i). Din relația (AP5.1.2) rezultă

$$\boldsymbol{A}_i\boldsymbol{y}\le\boldsymbol{0} \Leftrightarrow -\boldsymbol{A}\boldsymbol{y}\le\boldsymbol{0} \Leftrightarrow \begin{cases} y_2\ge 0 \\ y_1\ge y_3 \end{cases} \qquad \text{(AP5.1.7)}$$

astfel că există vectori $\boldsymbol{y}\in\mathbb{Z}^3$, $\boldsymbol{y}>\boldsymbol{0}$, care satisface (AP5.1.7) (de exemplu $\boldsymbol{y}=\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$), deci rețeaua $\boldsymbol{N}_i$ *este structural mărginită*.
(ii). Rețeaua $\boldsymbol{N}_i$ *nu este conservativă*, dar *este parțial conservativă*, ca şi rețeaua $\boldsymbol{N}$.
(iii). Din (AP5.1.4) rezultă

$$\boldsymbol{A}_i^T\boldsymbol{x}\ge\boldsymbol{0} \Leftrightarrow -\boldsymbol{A}^T\boldsymbol{x}\ge\boldsymbol{0} \Leftrightarrow \begin{cases} x_1=x_2 \\ x_2\le 0 \end{cases} \qquad \text{(AP5.1.8)}$$

deci rețeaua $\boldsymbol{N}_i$ *nu este repetitivă* şi nici *parțial repetitivă*.
(iv). Rețeaua $\boldsymbol{N}_i$ *nu este* nici *consistentă*, nici *parțial consistentă*, ca şi rețeaua $\boldsymbol{N}$.

3. Invarianții de tip *P* (*T*) ai rețelei $\boldsymbol{N}_i$ coincid cu cei ai rețelei $\boldsymbol{N}$. Deci orice vector de forma $\boldsymbol{y}=\begin{bmatrix}\alpha & 0 & \alpha\end{bmatrix}^T$, cu $\alpha\in\mathbb{Z}$, $\alpha>0$, este un invariant de tip *P* al rețelei $\boldsymbol{N}_i$ şi rețeaua $\boldsymbol{N}_i$ nu are nici un invariant *T*.

C. Rețeaua $\boldsymbol{N}_d$ din fig. AP5.1.1.(c) este *duala* rețelei $\boldsymbol{N}$ din fig. AP5.1.1.(a).
1. Matricele de incidență corespunzătoare lui $\boldsymbol{N}_d$ sunt date de:

$$\boldsymbol{A}_d^+ = (\boldsymbol{A}^-)^T, \quad \boldsymbol{A}_d^- = (\boldsymbol{A}^+)^T \;\Rightarrow\; \boldsymbol{A}_d = -\boldsymbol{A}^T. \qquad \text{(AP5.1.9)}$$

2.(i). Din relaţia (AP5.1.4) rezultă că $\boldsymbol{N}_d$ este structural mărginită deoarece $\boldsymbol{N}$ este structural repetitivă.
(ii). Din relaţia (AP5.1.5) rezultă că $\boldsymbol{N}_d$ nu este nici conservativă, nici parţial conservativă deoarece $\boldsymbol{N}$ nu este nici consistentă, nici parţial consistentă.
(iii). Din relaţia (AP5.1.2) rezultă că $\boldsymbol{N}_d$ este numai parţial repetitivă.
(iv). Din relaţia (AP5.1.3) rezultă că $\boldsymbol{N}_d$ este numai parţial consistentă.

3. Utilizând relaţia (AP5.1.9), rezultă că invarianţii de tip P (T) ai reţelei $\boldsymbol{N}_d$ coincid cu invarianţii de tip T (respectiv P) ai reţelei $\boldsymbol{N}$, astfel că $\boldsymbol{N}_d$ nu are invarianţi de tip P şi orice vector de forma $\boldsymbol{x} = \begin{bmatrix} \alpha & 0 & \alpha \end{bmatrix}^T$, cu $\alpha \in \mathbb{Z}$, $\alpha > 0$, este un invariant de tip T al reţelei $\boldsymbol{N}_d$.

D. Reţeaua $\boldsymbol{N}_{di}$ din fig. AP5.1.1.(d) este *duala inversată* a reţelei $\boldsymbol{N}$ din fig. AP5.1.1.(a), astfel că se obţin următoarele rezultate:
1. Matricele de incidenţă corespunzătoare lui $\boldsymbol{N}_{di}$ sunt date de:

$$\boldsymbol{A}_{di}^+ = (\boldsymbol{A}^+)^T, \quad \boldsymbol{A}_{di}^- = (\boldsymbol{A}^-)^T \;\Rightarrow\; \boldsymbol{A}_{di} = \boldsymbol{A}^T. \qquad \text{(AP5.1.10)}$$

2.(i). Din relaţia (AP5.1.8) rezultă că $\boldsymbol{N}_{di}$ nu este structural mărginită, deoarece $\boldsymbol{N}_i$ nu este nici repetitivă şi nici parţial repetitivă.
(ii). Ca şi reţeaua $\boldsymbol{N}_d$, nici $\boldsymbol{N}_{di}$ nu este conservativă şi nici parţial conservativă.
(iii). Reţeaua $\boldsymbol{N}_{di}$ nu este repetitivă.
(iv). Reţeaua $\boldsymbol{N}_{di}$ este parţial consistentă.

3. Utilizând relaţia (AP5.1.10) rezultă că invarianţii de tip P (T) ai reţelei $\boldsymbol{N}_{di}$ coincid cu invarianţii de tip T (respectiv P) ai reţelei $\boldsymbol{N}$, astfel că $\boldsymbol{N}_{di}$ nu are invarianţi de tip P şi orice vector de forma $\boldsymbol{x} = \begin{bmatrix} \alpha & 0 & \alpha \end{bmatrix}^T$, cu $\alpha \in \mathbb{Z}$, $\alpha > 0$, este un invariant de tip T al reţelei $\boldsymbol{N}_{di}$.

AP5.2

Ne plasăm în contextul modelului construit în AP4.1 punctul 1
1. Să se determine matricele de incidenţă $\boldsymbol{A}^-$, $\boldsymbol{A}^+$, $\boldsymbol{A}$ care descriu topologia acestui model.
2. Să se determine invarianţii P fundamentali ai reţelei şi să se precizeze semnificaţia fizică a acestora.
3. Să determine invarianţii T fundamentali ai reţelei şi să se precizeze semnificaţia fizică a lor.
4. Să se investigheze următoarele proprietăţi structurale:
   (i) mărginirea structurală;
   (ii) conservativitatea;
   (iii) repetitivitatea;
   (iv) consistenţa.
5. Să se precizeze semnificaţia fizică a rezultatelor de la punctul 4.

Soluție

1. Modelul de tip rețea Petri netemporizată a structurii de conducere care realizează o excludere mutuală a alocării resurselor partajate a fost obținut în capitolul precedent, aplicația **AP4.1**, cu ajutorul algoritmului de sinteză hibridă. Rețeaua rezultată este prezentată în fig. AP4.1.2.

Matricele $\boldsymbol{A}^-$, $\boldsymbol{A}^+$, precum şi matricea de incidență $\boldsymbol{A}$, pot fi vizualizate cu ajutorul mediului ***Petri Net Toolbox*** utilizând comanda ***Incidence Matrix*** din meniul ***Properties***; rezultatele sunt prezentate în fig. AP5.2.1.

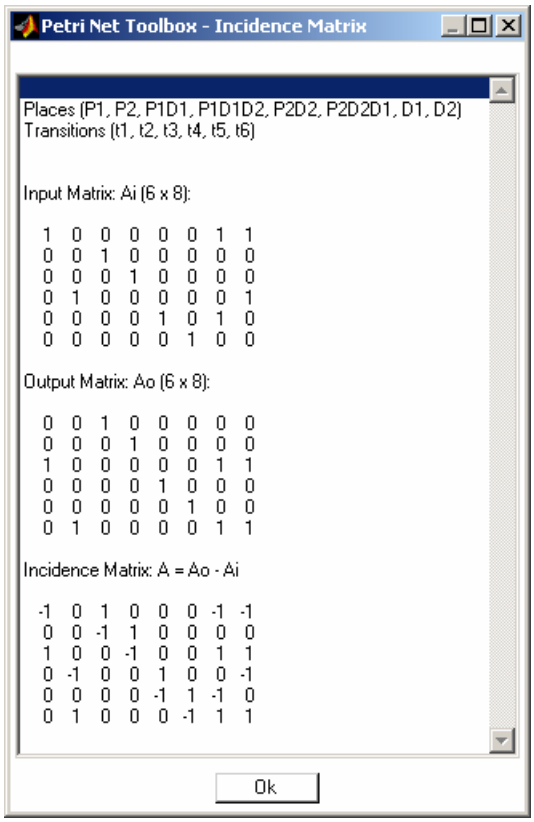


Fig. AP5.2.1. *Matricele de incidență pentru rețeaua Petri din fig. AP4.1.2.*

2. Matricea de incidență posedă rangul 4, deci rețeaua din fig. AP4.1.2. va avea maximum $m - \text{rang}(\boldsymbol{A}) = 8 - 4 = 4$ invarianți P liniar independenți.

Pornind de la semnificația fizică a invarianților P de bază, aceştia pot fi scrişi sub forma (ordinea în care sunt considerate pozițiile fiind cea din fig. AP5.2.1):

$$\boldsymbol{y}^1 = [1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0]^T,$$

$$\boldsymbol{y}^2 = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0]^T,$$

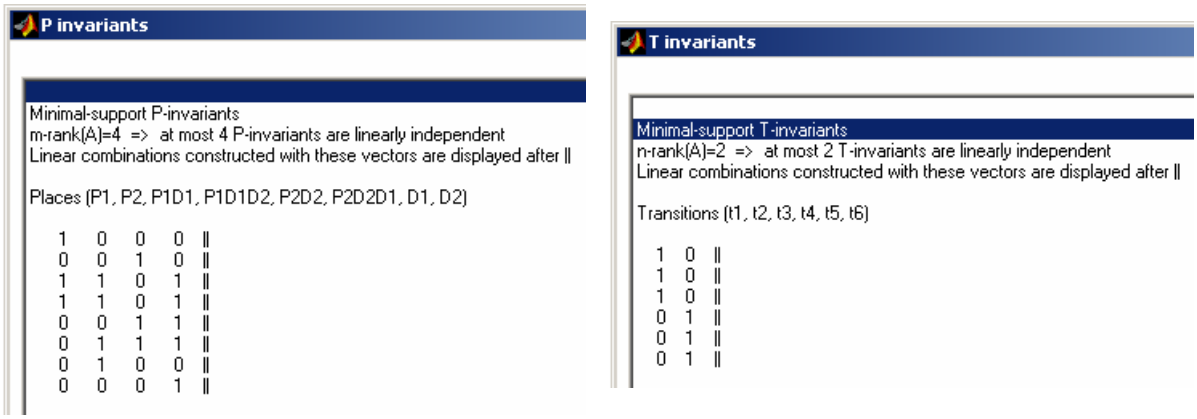
$$\boldsymbol{y}^3 = [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0]^T,$$

$$\boldsymbol{y}^4 = [0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 0 \quad 1]^T,$$

drept urmare a considerentelor de mai jos:

- Invariantul $\boldsymbol{y}^1$ este asociat procesorului P_1 şi arată că procesorul poate fi neutilizat (P1), utilizat pentru a crea fişier pe D_1 (P1D1), sau utilizat pentru a copia fişierul de pe D_1 pe D_2 (P1D1D2).
- Invariantul $\boldsymbol{y}^2$ este asociat procesorului P_2 şi arată că procesorul poate fi neutilizat (P2), utilizat pentru a crea fişier pe D_2 (P2D2), sau utilizat pentru a copia fişierul de pe D_2 pe D_1 (P2D2D1).
- Invariantul $\boldsymbol{y}^3$ este asociat discului D_1 şi arată că discul poate fi neutilizat (D1), utilizat pentru crearea unui fişier de către P_1 (P1D1), utilizat pentru ca P_1 să copieze de pe el pe D_2 (P1D1D2), sau utilizat pentru ca P_2 să copieze pe el de pe D_2 (P2D2D1).
- Invariantul $\boldsymbol{y}^4$ este asociat discului D_2 şi arată că discul poate fi neutilizat (D2), utilizat pentru crearea unui fişier de către P_2 (P2D2), utilizat pentru ca P_2 să copieze de pe el pe D_1 (P2D2D1), sau utilizat pentru ca P_1 să copieze pe el de pe D_1 (P1D1D2) sau alocat cu anticipare (P1D1) pentru prima operație efectuată de P_1.

Se observă că cei patru vectori, determinați mai sus din rațiuni de comportare fizică, satisfac ecuația de definiție a invarianților de tip *P*, $\boldsymbol{Ay = 0}$. De semenea, aceşti vectori sunt furnizați de mediul ***Petri Net Toolbox*** utilizând comanda ***P-Invariants*** din meniul ***Properties / Invariants*** - fig. AP5.2.2.(a).


P invariants
Minimal-support P-invariants
m-rank(A)=4 => at most 4 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places (P1, P2, P1D1, P1D1D2, P2D2, P2D2D1, D1, D2)
1 0 0 0 ||
0 0 1 0 ||
1 1 0 1 ||
1 1 0 1 ||
0 0 1 1 ||
0 1 1 1 ||
0 1 0 0 ||
0 0 0 1 ||
T invariants
Minimal-support T-invariants
m-rank(A)=2 => at most 2 T-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Transitions (t1, t2, t3, t4, t5, t6)
1 0 ||
1 0 ||
1 0 ||
0 1 ||
0 1 ||
0 1 ||

(a) (b)

Fig. AP5.2.2. *Invarianții fundamentali de tip P* (*a*) *şi de tip T* (*b*) *determinați de* ***Petri Net Toolbox*** *pentru rețeaua Petri din fig. AP4.1.2.*

3. Deoarece matricea de incidență posedă rangul 4, rețeaua din fig. AP4.1.2. va avea maximum $n - \text{rang}(\boldsymbol{A}) = 6 - 4 = 2$ invarianți *T* liniar independenți.

Pornind de la semnificația fizică a invarianților *T* de bază, aceştia pot fi scrişi sub forma (ordinea în care sunt considerate tranzițiile fiind cea din fig. AP5.2.1):

$$\boldsymbol{x}^1 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^T,$$

$$\boldsymbol{x}^2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T.$$

drept urmare a considerentelor de mai jos:

- Invariantul $\boldsymbol{x}^1$ este asociat secvenței de executări de tranziții $\sigma^1 : t_1, t_2, t_3$ ce corespunde servirii complete a unui task de tipul ST1, care porneşte din starea inițială cu toate resursele disponibile şi, în final, revine în această stare.

- Invariantul $\boldsymbol{x}^2$ este asociat secvenței de executări de tranziții $\sigma^2 : t_4, t_5, t_6$ ce corespunde servirii complete a unui task de tipul ST2, care porneşte din starea inițială cu toate resursele disponibile şi, în final, revine în această stare.

Se observă că cei doi vectori, determinați mai sus din rațiuni fizice, satisfac ecuația de definiție a invarianților de tip *T*, $\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}$. De semenea, aceşti vectori sunt furnizați de mediul ***Petri Net Toolbox*** utilizând comanda ***T-Invariants*** din meniul ***Properties / Invariants*** - fig. AP5.2.2.(b).

4. Se poate observa că rețeaua este acoperită atât de invarianți *P* cât şi de invarianți *T*. Utilizând **Teoremele 5.2.7** şi **5.2.8** putem spune că rețeaua din fig. AP4.1.2. este structural mărginită, conservativă, repetitivă şi consistentă, rezultate care sunt confirmate şi de ***Petri Net Toolbox*** (fig. AP5.2.3.).

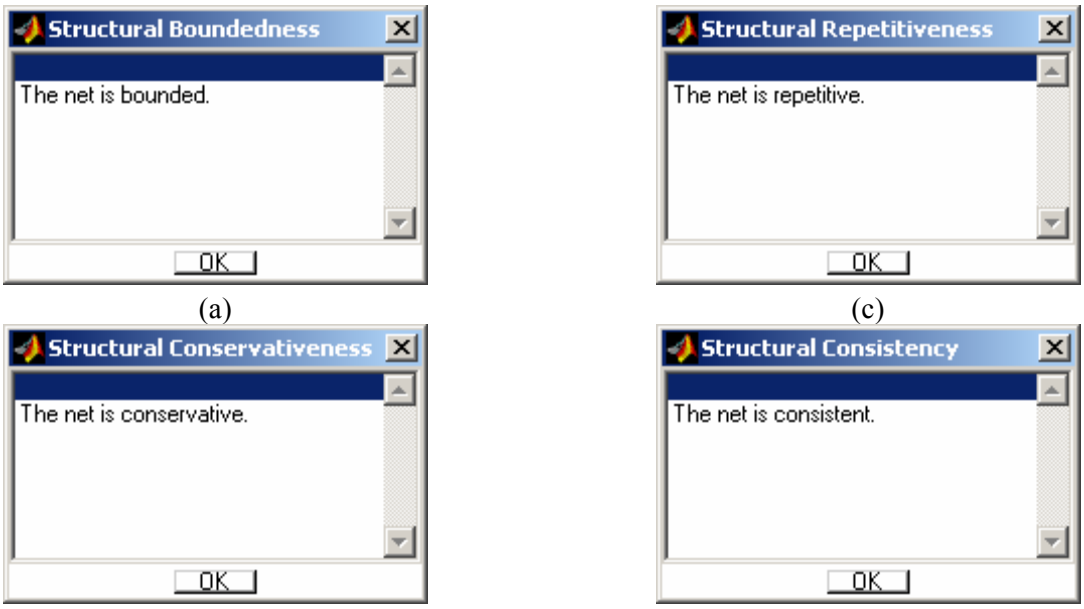


Fig. AP5.2.3. *Studierea proprietăților structurale cu ajutorul mediului* ***Petri Net Toolbox***.

5. Semnificația fizică a rezultatelor de la punctul anterior este următoarea:

- *Mărginirea structurală* ne arată că, oricât de mare ar fi numărul de resurse disponibile la momentul inițial şi oricât de multe operații ar fi in curs de servire (pentru orice marcaj inițial finit), în sistem nu se pot desfăşura un număr infinit de activități (deoarece în nici o poziție nu se pot acumula un număr infinit de jetoane);
- *Conservativitatea* ne arată că numărul total de resurse existente la momentul inițial (utilizate sau neutilizate) se conservă pe tot parcursul funcționării sistemului de calcul;
- *Repetitivitatea* indică faptul că există o configurație a resurselor disponibile şi a operațiilor în curs de servire, de la care plecând, toate activitățile specifice sistemului vor fi începute şi respectiv încheiate de un număr infinit de ori;

- *Consistenţa* indică faptul că există o configuraţie a resurselor disponibile şi a operaţiilor în curs de servire, de la care plecând, există o succesiune de evenimente însemnând începerea şi respectiv încheierea tuturor activităţilor (cel puţin o dată), ce permite revenirea la starea iniţială.

AP5.3

Ne plasăm în contextul modelului construit în **AP4.2** punctul 1.(ii).

1. Să se determine matricele de incidenţă $\boldsymbol{A}^-$, $\boldsymbol{A}^+$, $\boldsymbol{A}$.
2. Să se determine invarianţii P fundamentali ai reţelei şi să se precizeze semnificaţia fizică a acestora.
3. Să determine invarianţii T fundamentali ai reţelei şi să se precizeze semnificaţia fizică a lor.
4. Să se investigheze următoarele proprietăţi structurale:
   (i) mărginirea structurală;
   (ii) conservativitatea;
   (iii) repetitivitatea;
   (iv) consistenţa.
5. Să se precizeze semnificaţia fizică a rezultatelor de la punctul 4.

Soluţie

1. Modelul de tip reţea Petri netemporizată a structurii de conducere care realizează o excludere mutuală a alocării resursei partajate a fost obţinut în capitolul precedent, aplicaţia **AP4.2**, cu ajutorul algoritmului de sinteză hibridă. Reţeaua rezultată este prezentată în fig. AP4.2.5.

Matricea de incidenţă $\boldsymbol{A}$, precum şi matricele $\boldsymbol{A}^-$, $\boldsymbol{A}^+$, obţinute cu ajutorul mediului ***Petri Net Toolbox*** sunt prezentate în fig. AP5.3.1.

2. Matricea de incidenţă posedă rangul 5, deci reţeaua din fig. AP4.2.5 va avea maximum $m - \text{rang}(\boldsymbol{A}) = 10 - 5 = 5$ invarianţi P liniar independenţi.

Pornind de la semnificaţia fizică a invarianţilor P de bază, aceştia pot fi scrişi sub forma (ordinea în care sunt considerate poziţiile fiind cea din fig. AP5.3.1):

$$\boldsymbol{y}^1 = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\end{bmatrix}^T,$$

$$\boldsymbol{y}^2 = \begin{bmatrix}0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix}^T,$$

$$\boldsymbol{y}^3 = \begin{bmatrix}0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0\end{bmatrix}^T,$$

$$\boldsymbol{y}^4 = \begin{bmatrix}0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1\end{bmatrix}^T,$$

$$\boldsymbol{y}^5 = \begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0\end{bmatrix}^T.$$

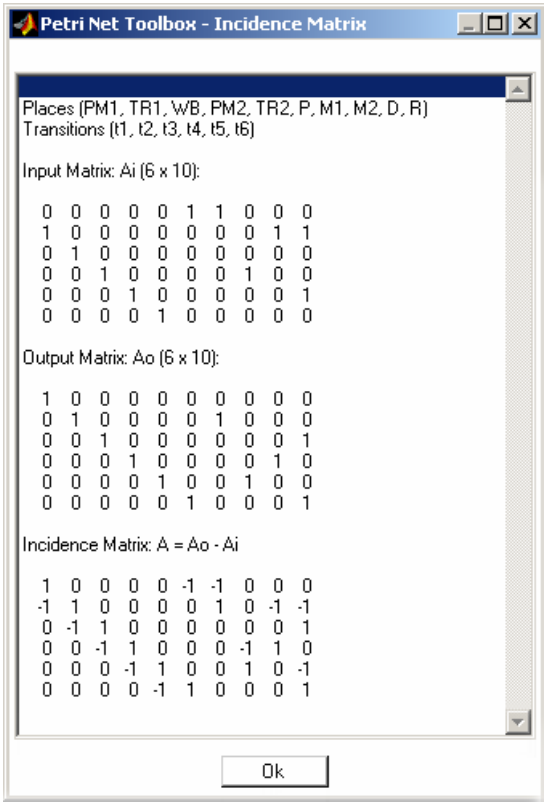


Fig. AP5.3.1. *Matricele de incidență pentru rețeaua Petri din fig. AP4.2.5.*

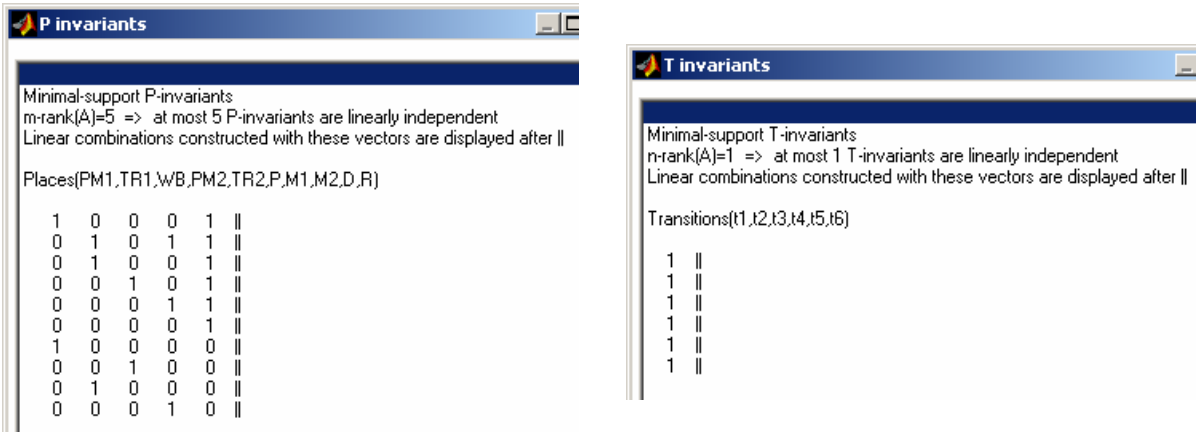
Semnificația fizică a invarianților *P* de bază ai rețelei din fig. AP4.2.5 este următoarea:

- Invariantul $\boldsymbol{y}^1$ este asociat mașinii M_1 și arată că mașina poate fi neutilizată (M1), sau utilizată (PM1).
- Invariantul $\boldsymbol{y}^2$ este asociat depozitului D și arată că locurile din depozit pot fi libere (D), ocupate (WB) sau alocate cu anticipare (TR1).
- Invariantul $\boldsymbol{y}^3$ este asociat mașinii M_2 și arată că mașina poate fi neutilizată (M2) sau utilizată (PM2).
- Invariantul $\boldsymbol{y}^4$ este asociat robotului R și arată că robotul poate fi neutilizat (R), utilizat pentru descărcarea mașinii M_1 (TR1), sau utilizat pentru descărcarea mașinii M_2 (TR2).
- Invariantul $\boldsymbol{y}^5$ este asociat paletelor P și arată că paletele pot fi neutilizate (P), sau se pot afla încărcate cu piese în una din pozițiile PM1, TR1, WB, PM2, TR2.

Se observă că cei cinci vectori, determinați mai sus din rațiuni fizice, satisfac ecuația de definiție a invarianților *P*, $\boldsymbol{Ay} = \boldsymbol{0}$. De semenea, acești vectori sunt furnizați de mediul ***Petri Net Toolbox*** - fig. AP5.3.2.(a).

3. Deoarece matricea de incidență posedă rangul 5, rețeaua din fig. AP4.1.2. va avea maximum $n - \text{rang}(\boldsymbol{A}) = 6 - 5 = 1$ invariant *T* liniar independent.

Pornind de la semnificația fizică a invarianților *T* de bază, acesta poate fi scris sub forma: $x = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$.


P invariants
Minimal-support P-invariants
m-rank(A)=5 => at most 5 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places(PM1,TR1,WB,PM2,TR2,P,M1,M2,D,R)
1 0 0 0 1 ||
0 1 0 1 1 ||
0 1 0 0 1 ||
0 0 1 0 1 ||
0 0 0 1 1 ||
0 0 0 0 1 ||
1 0 0 0 0 ||
0 0 1 0 0 ||
0 1 0 0 0 ||
0 0 0 1 0 ||
T invariants
Minimal-support T-invariants
n-rank(A)=1 => at most 1 T-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Transitions(t1,t2,t3,t4,t5,t6)
1 ||
1 ||
1 ||
1 ||
1 ||
1 ||

(a) (b)

Fig. AP5.3.2. *Invarianţii fundamentali de tip P* (*a*) *şi de tip T* (*b*) *determinaţi de* ***Petri Net Toolbox*** *pentru reţeaua Petri din fig. AP4.2.5.*

Invariantul $\boldsymbol{x}$ este asociat secvenţei de executări de tranziţii $\sigma : t_1, t_2, t_3, t_4, t_5, t_6$ corespunzătoare prelucrării complete a unei piese brute, care porneşte din starea iniţială cu toate resursele disponibile şi, în final, revine în această stare.

Se observă că vectorul $\boldsymbol{x}$, determinat mai sus din raţiuni fizice, satisface ecuaţia de definiţie a invarianţilor *T*, $\boldsymbol{A^T x = 0}$. De semenea, acest vector este furnizat de mediul ***Petri Net Toolbox*** - fig. AP5.3.2.(b).

4. Se poate observa că reţeaua este acoperită atât de invarianţi *P* cât şi de invarianţi *T*. Utilizând **Teoremele 5.2.7** şi **5.2.8** putem spune că reţeaua din fig. AP4.2.5 este structural mărginită, conservativă, repetitivă şi consistentă, rezultate care sunt confirmate şi de ***Petri Net Toolbox***.

5. Pe seama modului de tratare ilustrat în **AP5.2** sugerăm cititorului să comenteze semnificaţiile fizice ale proprietăţilor structurale analizate la punctul anterior.

AP5.4

Două task-uri partajează aceeaşi unitate centrală după algoritmul *Round-Robin* (RR) astfel: unitatea centrală execută o secvenţă de instrucţiuni din taskul 1 şi apoi trece la execuţia unei secvenţe de instrucţiuni din taskul 2, după care revine şi execută următoarea secvenţă de instrucţiuni din taskul 1, comutând apoi la următoarea secvenţă de instrucţiuni din taskul 2 ş.a.m.d. – adaptare după (David et Alla, 1992).

1. Construiţi modelul de tip reţea Petri al sistemului a cărui funcţionare decurge după algoritmul RR, în următoarele condiţii:
   (i) la fiecare servire a unui task se execută câte o singură instrucţiune din taskul respectiv.

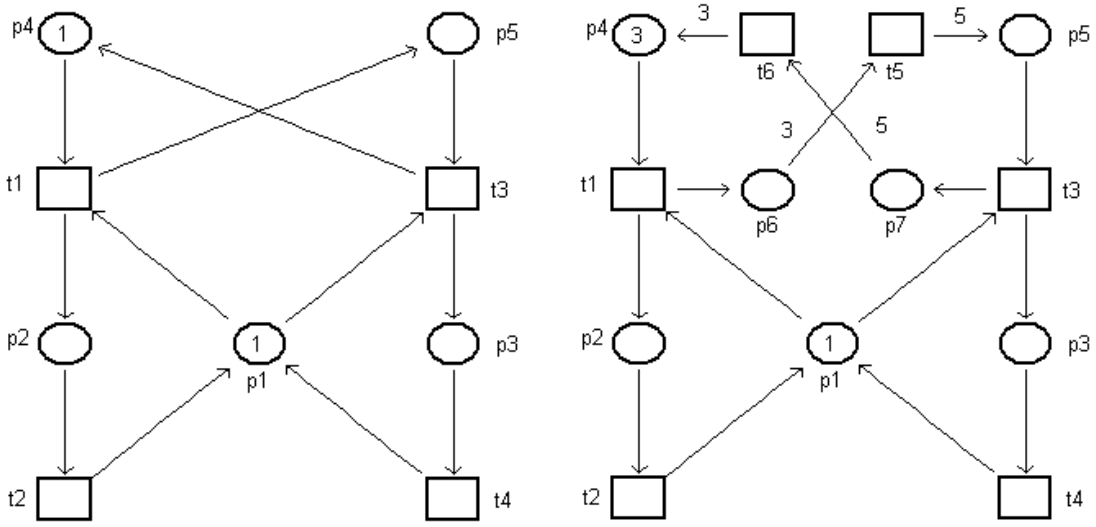
(ii) la fiecare servire a taskului 1 se execută secvențial un set de trei instrucțiuni, iar la fiecare servire a taskului 2 se execută secvențial un set de cinci instrucțiuni.

Pentru fiecare din modelele construite, precizați semnificația pozițiilor şi tranzițiilor.

2. Determinați invarianții P şi T de bază pentru modelele construite la punctul 1.

Soluție

1. Modelele de tip rețea Petri ce descriu funcționarea unității centrale după algoritmul RR în condițiile de la punctele (i) şi (ii) sunt prezentate în fig. AP5.4.1.(a) şi, respectiv, AP5.4.1.(b). Semnificațiile fizice ale pozițiilor şi tranzițiilor din cele două modele este precizată în tabelele AP5.4.1 şi, respectiv, AP5.4.2.



(a) (b)

Fig. AP5.4.1. *Rețelele Petri care modelează servirea a două taskuri după algoritmul RR*
(a) *executând câte o singură instrucțiune din fiecare task*;
(b) *executând câte o secvență de 3 instrucțiuni din taskul 1 şi de 5 instrucțiuni din taskul 2*.

Tab. AP5.4.1. *Semnificațiile tranzițiilor şi pozițiilor din modelul prezentat în fig. AP5.4.1.*(*a*).

Tranziție	**Eveniment**
t_1	Începerea executării unei instrucțiuni din taskul 1
t_2	Terminarea executării unei instrucțiuni din taskul 1
t_3	Începerea executării unei instrucțiuni din taskul 2
t_4	Terminarea executării unei instrucțiuni din taskul 2

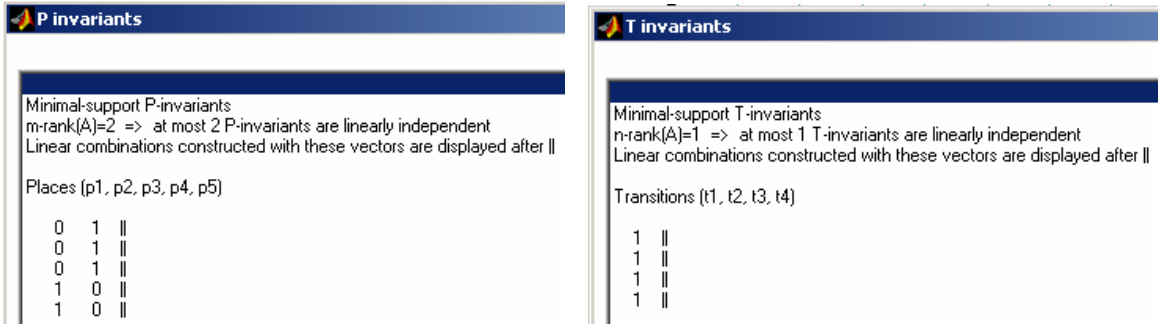
Poziție	**Condiție**
p_1	Unitatea centrală disponibilă
p_2	O instrucțiune din taskul 1 este în curs de execuție
p_3	O instrucțiune din taskul 2 este în curs de execuție
p_4	Procesorul poate fi alocat pentru o instrucțiune din taskul 1
p_5	Procesorul poate fi alocat pentru o instrucțiune din taskul 2

Tab. AP5.4.2. *Semnificațiile tranzițiilor şi pozițiilor din modelul prezentat în fig. AP5.4.1.*(*b*).

Tranziție	**Eveniment**
t_1	Începerea executării unei instrucțiuni din taskul 1
t_2	Terminarea executării unei instrucțiuni din taskul 1
t_3	Începerea executării unei instrucțiuni din taskul 2
t_4	Terminarea executării unei instrucțiuni din taskul 2
t_5	Preluarea controlul procesorului de taskul 2 de la taskul 1
t_6	Preluarea controlul procesorului de taskul 1 de la taskul 2

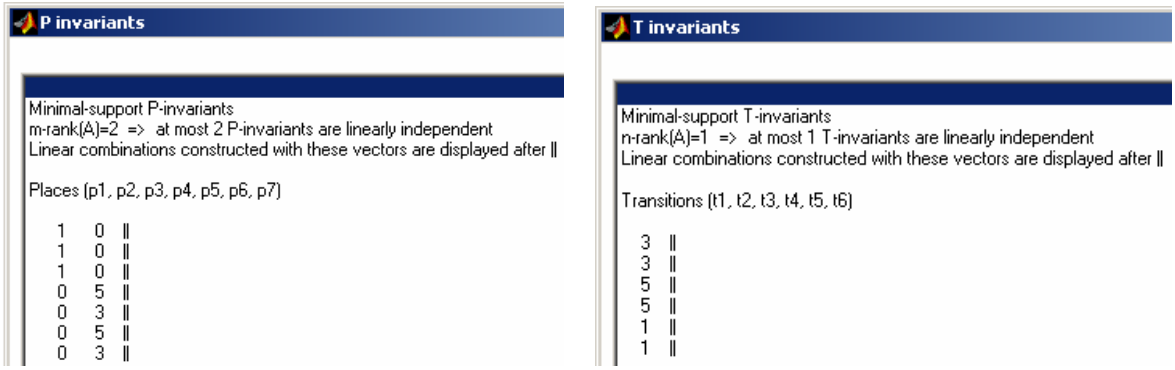
Poziție	**Condiție**
p_1	Unitatea centrală disponibilă
p_2	O instrucțiune din taskul 1 este în curs de execuție
p_3	O instrucțiune din taskul 2 este în curs de execuție
p_4	Procesorul poate fi alocat pentru o instrucțiune din taskul 1
p_5	Procesorul poate fi alocat pentru o instrucțiune din taskul 2
p_6	Procesorul a fost alocat pentru o instrucțiune din taskul 1
p_7	Procesorul a fost alocat pentru o instrucțiune din taskul 2

2. Invarianții fundamentali de tip *P* şi de tip *T* determinați de ***Petri Net Toolbox*** pentru rețelele Petri din fig. AP5.4.1 sunt prezentați în fig. AP5.4.2 şi, respectiv, în fig. AP5.4.3. Cititorul este invitat să formuleze semnificația fizică a acestor invarianți.


P invariants
Minimal-support P-invariants
m-rank(A)=2 => at most 2 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places (p1, p2, p3, p4, p5)
0 1 ||
0 1 ||
0 1 ||
1 0 ||
1 0 ||
T invariants
Minimal-support T-invariants
n-rank(A)=1 => at most 1 T-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Transitions (t1, t2, t3, t4)
1 ||
1 ||
1 ||
1 ||

(a) (b)

Fig. AP5.4.2. *Invarianții fundamentali de tip P* (*a*) *şi de tip T* (*b*) *determinați de* ***Petri Net Toolbox*** *pentru rețeaua Petri din fig. AP5.4.1.*(*a*).


P invariants
Minimal-support P-invariants
m-rank(A)=2 => at most 2 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places (p1, p2, p3, p4, p5, p6, p7)
1 0 ||
1 0 ||
1 0 ||
0 5 ||
0 3 ||
0 5 ||
0 3 ||
T invariants
Minimal-support T-invariants
n-rank(A)=1 => at most 1 T-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Transitions (t1, t2, t3, t4, t5, t6)
3 ||
3 ||
5 ||
5 ||
1 ||
1 ||

(a) (b)

Fig. AP5.4.3. *Invarianții fundamentali de tip P* (*a*) *şi de tip T* (*b*) *determinați de* ***Petri Net Toolbox*** *pentru rețeaua Petri din fig. AP5.4.1.*(*b*).

Capitolul 6

Modele de tip rețea Petri cu temporizare deterministă

Breviar teoretic

BT6.1. Rețele cu tranziții temporizate

Se spune că o rețea Petri este *cu tranziții temporizate* sau *temporizată T*, dacă fiecărei tranziții t_i, i=1, ..., n, i se asociază un interval de timp $d_i \geq 0$, prin intermediul unei funcții de temporizare tip *T* (tranziție).

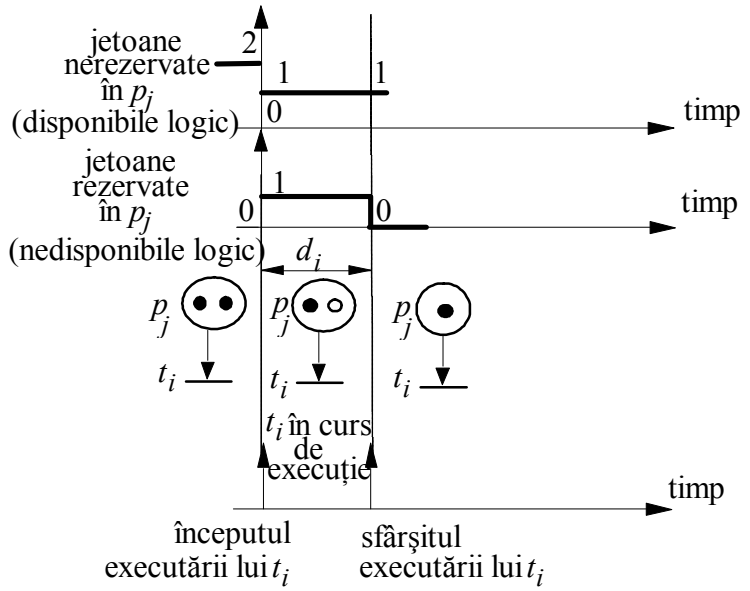


Fig. BT6.1.1. *Ilustrarea executării unei tranziții temporizate.*

În ceea ce privește funcționarea rețelei Petri temporizate *T*, intervalele de timp $d_i \geq 0$ joacă rolul unor *întârzieri* ce se manifestă după cum urmează: din momentul când tranziția t_i este validată, un număr de a_{ij}^- jetoane vor rămâne *rezervate* (nedisponibile din punctul de vedere calitativ, logic, al aplicării regulii tranziției) în poziția p_j care precede t_i pentru d_i unități de timp, înainte de deplasarea lor prin executarea tranziției t_i. (Reamintim că a_{ij}^-

notează ponderea arcului de la poziţia p_j la tranziţia t_i, fiind elementul generic al matricei de incidenţă de intrare $\boldsymbol{A}^-$). Ilustrăm cele spuse mai sus prin reprezentarea grafică din fig. BT6.1.1 pentru $a_{ij}^- = 1$ şi $M_0(p_j) = 2$. Jetonul rezervat este figurat ca un cerc, iar cel nerezervat ca un disc.

În cazul a două sau mai multe tranziţii aflate în conflict, selectarea tranziţiei care se va executa se realizează pe baza unui mecanism de priorităţi sau probabilităţi asignate respectivelor tranziţii. De asemenea, se poate folosi o structură de control de tipul celor prezentate în capitolul 4.

Atragem atenţia asupra faptului că duratele de timp asociate tranziţiilor nu joacă nici un rol în rezolvarea situaţiilor conflictuale, în sensul că nu este obligatoriu să se execute tranziţia corespunzătoare celei mai mici durate de timp.

BT6.2. Reţele cu poziţii temporizate

Se spune că o reţea Petri este *cu poziţii temporizate*, sau *temporizată P*, dacă fiecărei poziţii p_j, j = 1,...,m, i se asociază un interval de timp $d_j \geq 0$, prin intermediul unei funcţii de temporizare tip *P* (poziţie).

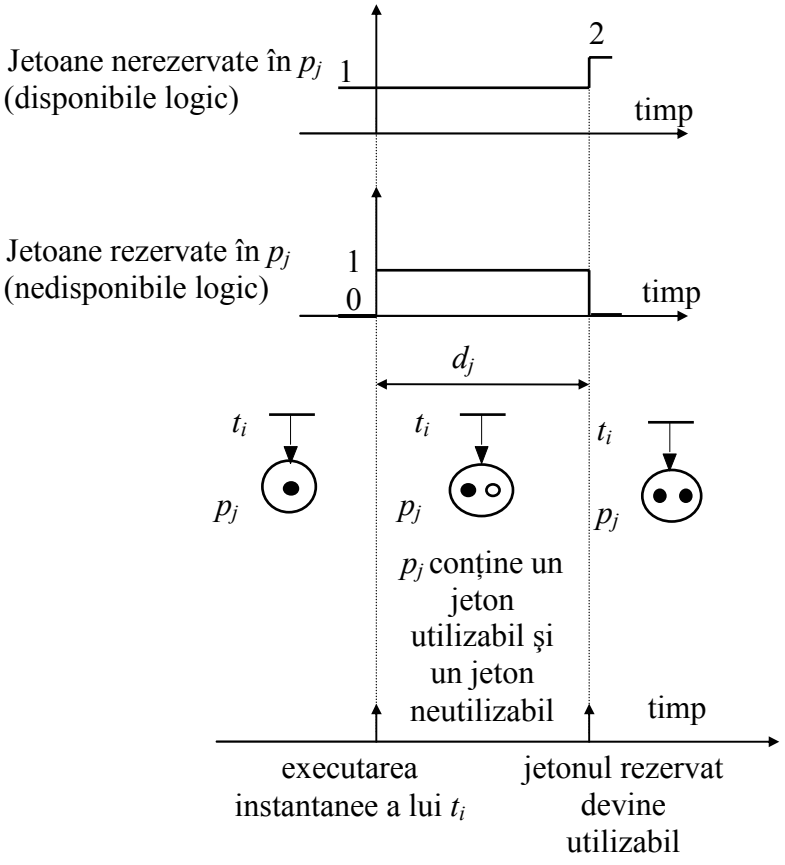


Fig. BT6.2.1. *Ilustrarea comportării unei poziţii temporizate.*

În ceea ce priveşte funcţionarea rețelei Petri temporizate P, intervalele de timp $d_j \geq 0$ joacă rolul unei întârzieri ce se manifestă după cum urmează: din momentul când tranziția t_i care precede p_j este executată, un număr de a_{ij}^+ jetoane vor rămâne rezervate (nedisponibile din punct de vedere calitativ, logic, al aplicării regulii tranziției) în poziția p_j pentru d_j unități de timp, înainte de a putea fi utilizate pentru a valida tranziții ce succed lui p_j. (Reamintim că a_{ij}^+ notează ponderea arcului de la tranziția t_i la poziția p_j, fiind elementul generic al matricei de incidență de ieşire $\boldsymbol{A}^+$).

În cazul temporizării P, se presupune că executarea oricărei tranziții validate are loc instantaneu (nu consumă timp). Ilustrăm cele spuse mai sus prin reprezentarea grafică din fig. BT6.2.1 pentru $a_{ij}^+ = 1$ şi $M_0(p_j) = 1$. Jetonul rezervat este figurat ca un cerc, iar cel nerezervat ca un disc.

În cazul a două sau mai multe tranziții aflate în conflict, selectarea tranziției care se va executa se realizează pe baza unui mecanism de priorități sau probabilități asignate respectivelor tranziții. De asemenea, se poate folosi o structură de control de tipul celor prezentate în capitolul 4.

Rețelele temporizate P se folosesc pentru a introduce în modelul matematic informațiile privitoare la durata activităților, elaborând, astfel, un model *cantitativ*. Cu ajutorul unui asemenea model se pot lua în discuție toate caracteristicile temporale specifice sosirii clienților şi servirii acestora cu anumite succesiuni de operații de către resursele sistemului.

În cazul unui model ce reprezintă o structură de conducere a unui sistem cu evenimente discrete, rezultat prin procedeul de sinteză hibridă prezentat în capitolul 4, fiecărei poziții asociate unei operații i se asignează durata activității de servire (procesare), iar fiecărei poziții asociate unei resurse i se asignează durata activității de eliberare a resursei.

BT6.3. Transformarea unei rețele temporizate *T* într-o rețea temporizată *P*

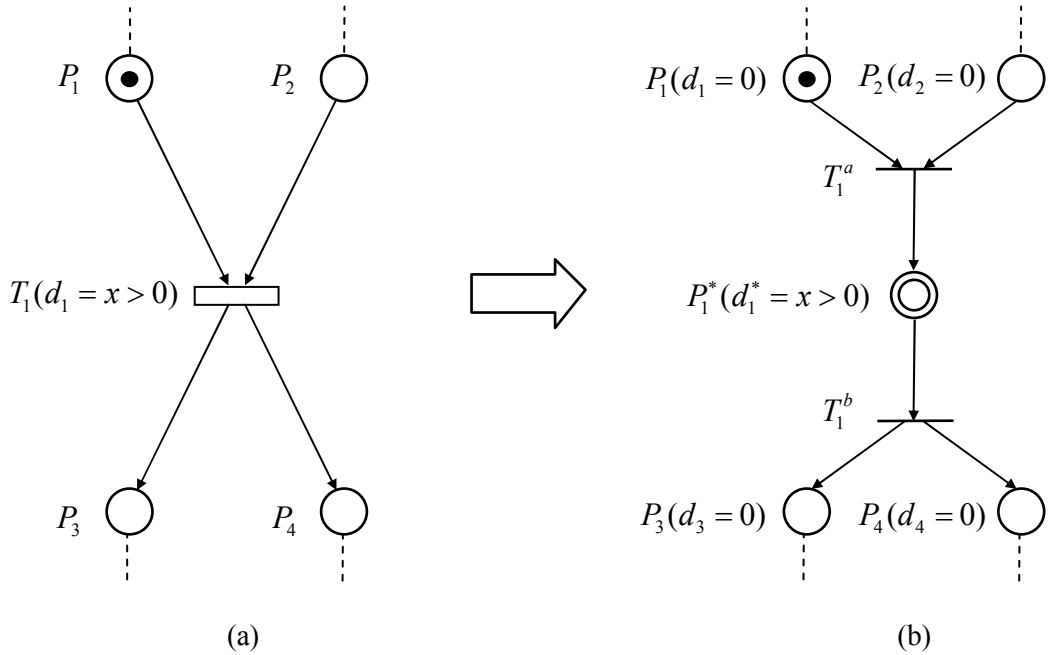
Pentru prezentarea procedeului de trecere de la temporizarea T la temporizarea P, utilizăm suportul grafic din fig. BT6.3.1. În fig. BT6.3.1.(a) este prezentată o subrețea dintr-o rețea temporizată T al cărei mod de operare este echivalent cu cel al rețelei temporizate P din fig. BT6.3.1.(b).

Tranziția temporizată T_1 este înlocuită prin structura echivalentă alcătuită din tranzițiile T_1^a, T_1^b şi poziția P_1^*. Toate pozițiile de intrare ale lui T_1 sunt poziții de intrare pentru T_1^a şi toate pozițiile de ieşire ale lui T_1 sunt poziții de ieşire pentru T_1^b. Durata de timp $d_1 = x > 0$ asignată tranziției T_1 în rețeaua temporizată T, va fi asociată, cu aceeaşi valoare (notată $d_1^* = x$) poziției P_1^* din rețeaua temporizată P. Acest procedeu se aplică pentru toate tranzițiile temporizate ale rețelei inițiale (temporizată T).

Pozițiile rețelei inițiale (temporizată T) sunt preluate identic în rețeaua rezultată (temporizată P) asignându-le durate nule (de exemplu, $d_i = 0,\ i = 1,2,3,4$, în fig. BT6.3.1.(b)). Duratele de timp sunt nenule numai pentru pozițiile suplimentare rezultate

în urma trecerii de la temporizarea *T* la temporizarea *P*. Pentru a realiza o distincție grafică ușor sesizabilă, pozițiile care au durate nenule sunt uneori figurate printr-un cerc dublu (ca în fig. BT6.3.1.(b)), spre deosebire de cercul simplu ce simbolizează pozițiile cu durată nulă.

Marcajul inițial al rețelei temporizate *T* este preluat identic în rețeaua rezultantă (temporizată *P*). Toate pozițiile suplimentare din rețeaua rezultată (temporizată *P*) care sunt introduse pentru a realiza temporizarea *P* (de exemplu P_1^* în fig. BT6.3.1.(b)) au marcajul inițial nul.



(a) (b)

Fig. BT6.3.1. *Ilustrarea procedeului de transformare a unei rețele temporizate T într-o rețea temporizată P*

(a) *rețeaua inițială* (*temporizată T*); (b) *rețeaua rezultată* (*temporizată P*).

BT6.4. Transformarea unei rețele temporizate *P* într-o rețea temporizată *T*

Pentru prezentarea procedeului de trecere de la temporizarea *P* la temporizarea *T*, utilizăm suportul grafic din fig. BT6.4.1. În fig. BT6.4.1.(a) este prezentată o subrețea dintr-o rețea temporizată *P* al cărui mod de operare este echivalent cu cel al rețelei temporizate *T* din fig. BT6.4.1.(b).

Poziția temporizată P_1 este înlocuită prin structura echivalentă alcătuită din pozițiile P_1^a, P_1^b şi tranziția T_1^*. Toate tranzițiile de intrare a lui P_1 sunt tranziții de intrare pentru P_1^a şi toate tranzițiile de ieşire ale lui P_1 sunt tranziții de ieşire pentru P_1^b. Durata de timp $d_1 = x > 0$ asignată poziției P_1 în rețeaua temporizată *P*, va fi asociată, cu aceeaşi valoare (notată $d_1^* = x > 0$) tranziției T_1^* din rețeaua temporizată *T*. Acest procedeu se aplică pentru toate pozițiile temporizate alei rețelei inițiale (temporizată *P*).

Tranzițiile rețelei inițiale (temporizată *P*) sunt preluate în mod identic în rețeaua rezultată (temporizată *T*) asignându-le durate nule (de exemplu $d_i = 0,\ i = 1,2,3,4$, în fig. BT6.4.1.(b)). Duratele de timp sunt nenule numai pentru tranzițiile suplimentare rezultate în urma trecerii de la temporizarea *P* la temporizarea *T*. Pentru a realiza o distincție grafică ușor sesizabilă, tranzițiile care au durate nenule sunt uneori figurate printr-un dreptunghi (ca în fig. BT6.4.1.(b)), spre deosebire de bara ce simbolizează tranzițiile cu durată nulă.

Marcajul inițial al rețelei rezultate (temporizată *T*) se alocă pe baza marcajului inițial al rețelei temporizate *P*, plasând jetoanele fie în pozițiile cu indice superior '*a*', fie în pozițiile cu indicele superior '*b*' (de exemplu P_1^b, în fig. BT6.4.1.(b)).

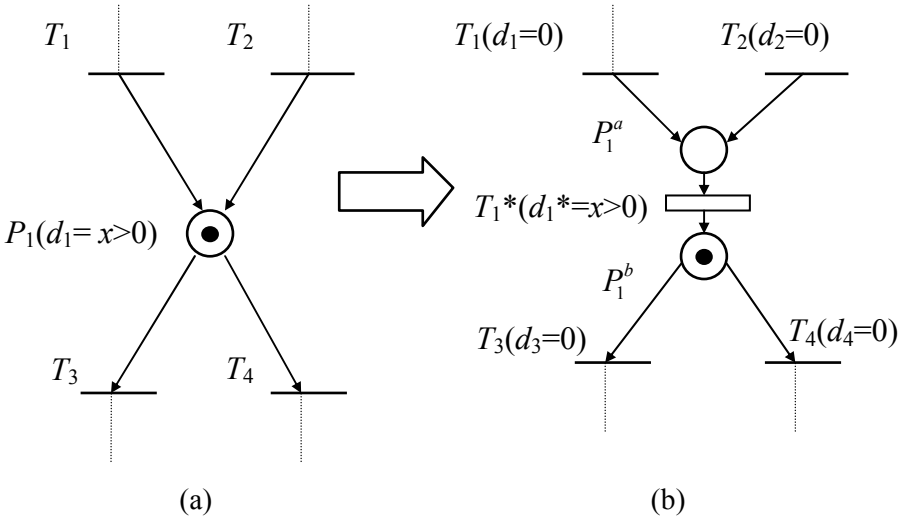


Fig. BT6.4.1. *Ilustrarea procedeului de transformare a unei rețele temporizate P într-o rețea temporizată T*
(a) *rețeaua inițială* (*temporizată P*)*;* (b) *rețeaua rezultată* (*temporizată T*).

BT6.5. Comportarea periodică a rețelelor Petri acoperite de invarianți *P*, (parțial) consistente, temporizate *T*

Fie ***N*** o rețea Petri acoperită de invarianți *P*, (parțial) consistentă, temporizată *T*, cu *m* poziții şi *n* tranziții, pentru care $\boldsymbol{A}^-$ notează matricea de incidență de intrare. Fie M_0 un marcaj inițial şi σ o secvență de executări de tranziții care satisfac definiția proprietății de (parțială) consistență a rețelei (adică rețeaua evoluează din marcajul inițial M_0 şi revine înapoi la M_0). Rețeaua prezintă o *comportare ciclică* caracterizată prin *durata unui ciclu* τ, care reprezintă intervalul de timp necesar executării secvenței σ. Această comportare ciclică are semnificația unui regim permanent în operarea rețelei, iar τ joacă rolul de *perioadă*.

Date fiind duratele $d_i \geq 0,\ i = 1,\ldots,n$, asociate tranzițiilor rețelei Petri, să construim matricea diagonală $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ care posedă pe diagonală valorile d_i:

$$\left(\boldsymbol{D}\right)_{ij} = \begin{cases} d_i, & i = j, \\ 0, & i \neq j, \end{cases} \quad i, j = 1, \ldots, n. \tag{BT6.5.1}$$

Să notăm prin $\boldsymbol{y}_k$, $k = 1, \ldots, p$, invarianţii P fundamentali ai reţelei. În termenii acestor construcţii prealabile, putem caracteriza durata unui ciclu prin:

Teorema 6.5.1. Durata unui ciclu satisface inegalitatea:

$$\tau \geq \tau_{\min} = \max_{k=\overline{1,p}} \left\{ \frac{\boldsymbol{y}_k^{\mathrm{T}} \left(\boldsymbol{A}^{-}\right)^{\mathrm{T}} \boldsymbol{D}\boldsymbol{x}}{\boldsymbol{y}_k^{\mathrm{T}} \boldsymbol{M}_0} \right\}, \tag{BT6.5.2}$$

unde $\boldsymbol{x} = \bar{\sigma}$ notează acel invariant T care corespunde secvenţei de executări $M_0[\sigma\rangle M_0$.

Observaţie: **Teorema 6.5.1** nu garantează că valoarea $\tau_{\min}$ este efectiv atinsă de perioada cu care operează reţeaua.

Caz particular: *Comportarea periodică a grafurilor marcate tare conexe, temporizate T*

Topologia de graf marcat tare conex a lui $\boldsymbol{N}$ asigură faptul că soluţiile întregi pozitive ale ecuaţiei $\boldsymbol{A}^T \boldsymbol{x} = \boldsymbol{0}$ sunt de forma $\boldsymbol{x} = \alpha [1 \ 1 \ \ldots \ 1]^T$ cu $\alpha \in \mathbb{N}^*$. Rezultă că $\boldsymbol{N}$ este consistentă, garantând existenţa unui marcaj iniţial M_0 şi a unei secvenţe de executări de tranziţii σ (care conţine fiecare tranziţie o singură dată, $\bar{\sigma} = [1 \ 1 \ \ldots \ 1]^T$) astfel încât putem scrie $M_0[\sigma\rangle M_0$. Pe de altă parte, poziţiile oricărui circuit elementar C_k, $k = 1, \ldots, p$, din $\boldsymbol{N}$ reprezintă suportul unui invariant P fundamental $\boldsymbol{y}_k$, $k = 1, \ldots, p$, pentru care toate elementele nenule au valoarea 1.

Coroborând toate aceste informaţii, putem afirma că operarea periodică a grafului marcat înseamnă executarea unei secvenţe de tranziţii în care toate tranziţiile grafului apar o singură dată, iar perioada este dată de următorul rezultat:

Teorema 6.5.2. Durata unui ciclu de funcţionare este:

$$\tau = \max_{k=\overline{1,p}} \left\{ \frac{D(C_k)}{M_0(C_k)} \right\}, \tag{BT6.5.3}$$

unde $D(C_k)$ şi $M_0(C_k)$ notează întârzierea totală pe circuitul C_k (suma tuturor întârzierilor pe tranziţiile lui C_k) şi, respectiv, numărul total de jetoane aflate pe circuitului C_k în marcajul iniţial.

Observaţie: Este evident că apariţia marcajului iniţial la numitorii fracţiilor din relaţiile (BT6.5.2) şi (BT6.5.3) determină scăderea duratei minime a ciclului de operare $\tau_{\min}$ atunci când conţinutul iniţial în jetoane creşte. Acest fapt este mai uşor de înţeles dacă se consideră frecvenţa maximă de operare $\nu_{\max} = 1/\tau_{\min}$ care reprezintă numărul maxim de executări ale secvenţei σ în unitatea de timp.

BT6.6. Comportarea periodică a rețelelor Petri acoperite de invarianți *P*, (parțial) consistente, temporizate *P*

Problematica abordată în acest paragraf este similară celei tratate în paragraful anterior, numai că în cazul de față considerăm o rețea Petri $\boldsymbol{N}$ acoperită de invarianți *P*, (parțial) consistentă, temporizată *P*, cu *m* poziții şi *n* tranziții, pentru care $\boldsymbol{A}^+$ notează matricea de incidență de ieşire. Fie M_0 un marcaj inițial şi σ o secvență de executări de tranziții care satisfac definiția proprietății de (parțială) consistență a rețelei (adică rețeaua evoluează din marcajul inițial M_0 şi revine înapoi la M_0). Rețeaua prezintă o *comportare ciclică* caracterizată prin *durata unui ciclu* τ, care reprezintă intervalul de timp necesar executării secvenței σ. Această comportare ciclică are semnificația unui regim permanent în operarea rețelei, iar τ joacă rolul de *perioadă*.

Date fiind duratele $d_j \geq 0,\ j = 1,\ldots,m$, asociate pozițiilor rețelei Petri să construim matricea diagonală $\boldsymbol{D} \in \mathbb{R}^{m\times m}$ care posedă pe diagonală valorile d_j:

$$\left(\boldsymbol{D}\right)_{ij} = \begin{cases} d_j, & i = j, \\ 0, & i \neq j, \end{cases} \quad i, j = 1,\ldots,m. \tag{BT6.6.1}$$

Să notăm prin $\boldsymbol{y}_k$, $k = 1,\ldots,p$, invarianții *P* fundamentali ai rețelei. În termenii acestor construcții prealabile, putem caracteriza durata unui ciclu prin:

Teorema 6.6.1. Durata unui ciclu satisface inegalitatea:

$$\tau \geq \tau_{\min} = \max_{k=1,p} \left\{ \frac{\boldsymbol{y}_k^{\mathrm{T}} \boldsymbol{D} \left(\boldsymbol{A}^+\right)^{\mathrm{T}} \boldsymbol{x}}{\boldsymbol{y}_k^{\mathrm{T}} \boldsymbol{M}_0} \right\}, \tag{BT6.6.2}$$

unde $\boldsymbol{x} = \bar{\sigma}$ notează acel invariant *T* care corespunde secvenței de executări $M_0[\sigma\rangle M_0$ (ilustrare în **AP6.3**, **AP6.4**).

Observație: **Teorema 6.6.1** nu garantează că valoarea $\tau_{\min}$ este efectiv atinsă de perioada cu care operează rețeaua (ilustrare în **AP6.5**).

Caz particular: *Comportarea periodică a grafurilor marcate tare conexe, temporizate P*

Corespunzător **Teoremei BT6.5.2** din paragraful anterior, pentru un graf marcat tare conex, temporizat *P*, se obține următorul rezultat:

Teorema 6.6.2. Durata unui ciclu de funcționare este:

$$\tau_{\min} = \max_{k=1,p} \left\{ \frac{D(C_k)}{M_0(C_k)} \right\}, \tag{BT6.6.3}$$

unde $D(C_k)$ şi $M_0(C_k)$ notează întârzierea totală pe circuitul C_k (suma tuturor întârzierilor pe pozițiile lui C_k) şi, respectiv, numărul total de jetoane aflate pe circuitului C_k în marcajul inițial.

Observaţia din paragraful anterior cu privire la efectul marcajului iniţial asupra duratei minime a unui ciclu de operare a reţelei îşi păstrează valabilitatea (ilustrare în **AP6.1**, **AP6.2** şi **AP6.8**).

Aplicaţiile din acest capitol sunt orientate pe ilustrarea proprietăţilor reţelelor Petri cu temporizare deterministă de tip *P* deoarece în capitolele anterioare am utilizat convenţia ca desfăşurarea activităţilor (aşadar consumatorii de timp) să fie modelată cu ajutorul poziţiilor. Modul în care informaţiile temporale pot fi asignate tranziţiilor va fi ilustrat în capitolele 7 şi 8, ca o manieră complementară de prezentare a includerii timpului pentru analiza cantitativă a reţelelor Petri. Totodată precizăm că, din punct de vedere istoric, operarea cu temporizări nedeterministe s-a dezvoltat pornind de la convenţia că informaţiile temporale sunt asignate tranziţiilor. Pe de altă parte, transformările reţelelor temporizate *T* în reţele temporizate *P* şi invers arată că, în condiţiile modelării corecte, alegerea convenţiei pentru asignarea informaţiilor temporale nu influenţează concluziile rezultate din analiza cantitativă.

Aplicaţii

AP6.1.

Se consideră reţelele Petri cu temporizare *P* din fig. AP6.1.1. Duratele de timp asignate poziţiilor sunt precizate între paranteze. Pentru fiecare dintre cele două reţele să se abordeze următoarea problematică:

1. Să se constate instalarea unui regim de funcţionare periodic şi să se precizeze perioada, utilizând în mod direct informaţiile temporale.
2. Să se verifice analitic rezultatul de la punctul 1 folosind **Teorema 6.6.2**.

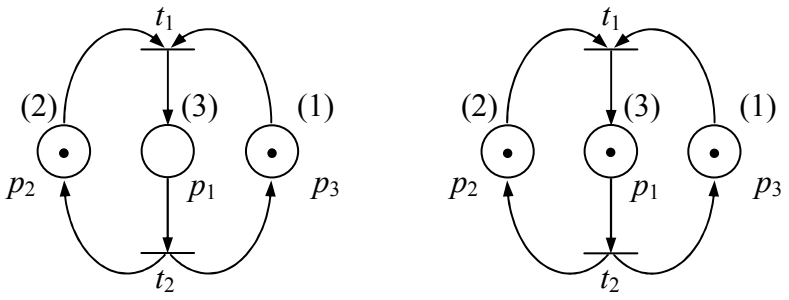


Fig. AP6.1.1. *Reţelele Petri utilizate în AP6.1.*

Soluţie

A. 1. Reţeaua din fig. AP6.1.1.(a) este graf marcat tare conex şi drept urmare, un ciclu de funcţionare a reţelei constă în executarea secvenţei $\sigma = t_1 t_2$ (care conţine fiecare tranziţie o singură dată) şi are proprietatea $M_0[\sigma\rangle M_0$. Se constată că tranziţia t_1 se execută la momentele

0, 5, 10, 15, ..., adică, generalizând, cea de-a *k*-a executare are loc la momentul $5(k-1)$, $k \in \mathbb{N}^*$. Analog, tranziția t_2 se execută la momentele 3, 8, 13, 18, ..., adică, generalizând, cea de-a *k*-a executare are loc la momentul $5(k-1)+3$, $k \in \mathbb{N}^*$. Rezultă că intervalul de timp necesar executării secvenței σ are durata 5, deci funcționarea ciclică a rețelei se desfăşoară cu perioada de 5 [unități de timp].

2. Pentru a aplica **Teorema 6.6.2**, identificăm circuitele elementare $C_1 = t_1 p_1 t_2 p_2$ şi $C_2 = t_1 p_1 t_2 p_3$, având $D(C_1) = 5$, $M_0(C_1) = 1$, şi, respectiv, $D(C_2) = 4$, $M_0(C_2) = 1$. Din relația (BT6.6.3) rezultă că durata unui ciclu de funcționare a rețelei din fig. AP6.1.1.(a) este egală cu 5 [unități de timp], valoare care a fost obținută şi la punctul 1 (utilizând în mod direct informațiile temporale).

B. 1. Rețeaua din fig. AP6.1.1.(b) are aceeaşi topologie ca şi cea din fig. AP6.1.1.(a), dar marcaj inițial diferit. Un ciclu de funcționare a rețelei constă în executarea secvenței $\sigma = t_1 t_2$. Tranziția t_1 se execută la momentele 0, 2, 5, 7, 10, 12, ..., adică, generalizând, cea de-a $(2k-1)$-a executare are loc la momentul $5(k-1)$, $k \in \mathbb{N}^*$, iar cea de-a $(2k)$-a executare are loc la momentul $5(k-1)+2$, $k \in \mathbb{N}^*$. Analog, tranziția t_2 se execută la momentele 0, 3, 5, 8, 10, 13, ..., adică, generalizând, cea de-a $(2k-1)$-a executare are loc la momentul $5(k-1)$, $k \in \mathbb{N}^*$, iar cea de-a $(2k)$-a executare are loc la momentul $5(k-1)+3$, $k \in \mathbb{N}^*$. Rezultă că într-un interval de 5 [unități de timp] secvența σ se execută de două ori, deci funcționarea ciclică a rețelei se desfăşoară cu perioada de 2,5 [unități de timp].

2. Pentru a aplica **Teorema 6.6.2**, identificăm circuitele elementare $C_1 = t_1 p_1 t_2 p_2$ şi $C_2 = t_1 p_1 t_2 p_3$, având $D(C_1) = 5$, $M_0(C_1) = 2$, şi, respectiv, $D(C_2) = 4$, $M_0(C_2) = 2$. Din relația (BT6.6.3) rezultă că durata unui ciclu de funcționare a rețelei din fig. AP6.1.1.(b) este egală cu 2,5 [unități de timp], valoare care a fost obținută şi la punctul 1 (utilizând în mod direct informațiile temporale).

Observație: Cele două rețele considerate în această aplicație au topologii identice, diferența constatată între perioadele lor de funcționare ciclică fiind datorată în exclusivitate marcajului inițial. În aceeaşi idee, recomandăm cititorului o comparație cu **AP9.2**.

AP6.2.

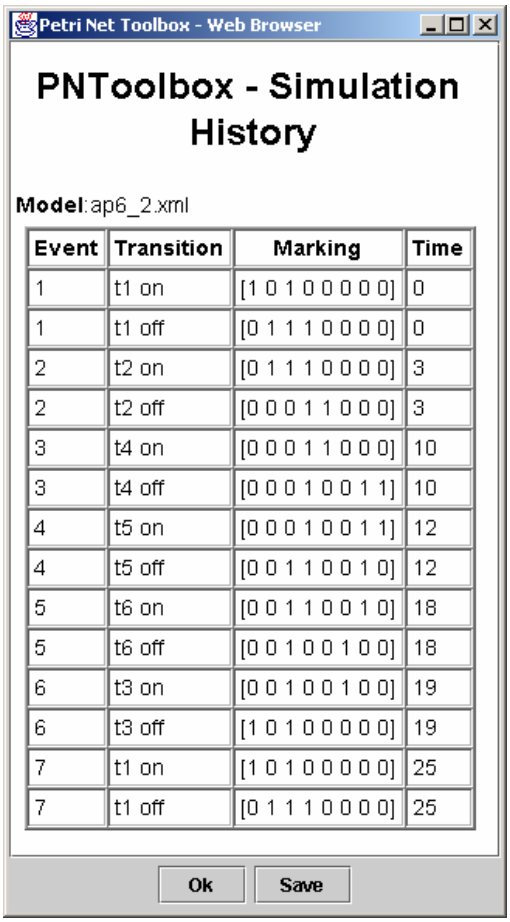
Fie protocolul de comunicații studiat în **AP2.4** şi **AP3.4**. Se temporizează modelul logic din fig. AP2.4.1 asignând fiecărei poziții p_i durata de timp $d(p_i)$ corespunzătoare operației pe care o reprezintă, pentru $i = 1, \ldots, 8$. Se consideră următoarele valori numerice: $d(p_1) = 6$, $d(p_2) = 3$, $d(p_3) = 5$, $d(p_4) = 4$, $d(p_5) = 7$, $d(p_6) = 1$, $d(p_7) = 8$ şi $d(p_8) = 2$ [unități de timp].

1. Să se constate instalarea unui regim de funcționare periodic şi să se precizeze perioada, utilizând în mod direct informațiile temporale.

2. Să se verifice analitic rezultatul de la punctul 1 folosind **Teorema 6.6.2**.
3. Să se precizeze modul de utilizare a mediului ***Petri Net Toolbox*** pentru a intra în posesia succesiunii de evenimente corespunzătoare transmiterii complete a unui mesaj şi a momentelor de timp la care au loc aceste evenimente.

Soluție

1. Rețeaua din fig. AP2.4.1 este graf marcat tare conex. Drept urmare, un ciclu de funcționare a rețelei constă în executarea unei secvenței σ care conține fiecare tranziție o singură dată şi are proprietatea $M_0[\sigma\rangle M_0$. Conform celor constatate la **AP2.4** în lipsa informațiilor temporale, există trei secvențe care satisfac cerințele de mai sus. Întrucât, pornind din marcajul inițial, contextul temporal utilizat face ca ordinea de executare a tranzițiilor să fie $\sigma = t_1 t_2 t_4 t_5 t_6 t_3$, ele se vor executa pentru prima dată la momentele 0, 3, 10, 12, 18, 19. Cunoscând momentele de timp la care se execută tranziția t_1, se constată că cea de-a k-a executare are loc la momentul $25(k-1)$, $k \in \mathbb{N}^*$. Rezultă că intervalul de timp necesar executării secvenței σ are durata 25, deci funcționarea ciclică a rețelei se desfăşoară cu perioada de 25 [unități de timp] în care are loc transmiterea completă a unui mesaj prin protocolul de comunicații modelat.



Event	Transition	Marking	Time
1	t1 on	[1 0 1 0 0 0 0 0]	0
1	t1 off	[0 1 1 1 0 0 0 0]	0
2	t2 on	[0 1 1 1 0 0 0 0]	3
2	t2 off	[0 0 0 1 1 0 0 0]	3
3	t4 on	[0 0 0 1 1 0 0 0]	10
3	t4 off	[0 0 0 1 0 0 1 1]	10
4	t5 on	[0 0 0 1 0 0 1 1]	12
4	t5 off	[0 0 1 1 0 0 1 0]	12
5	t6 on	[0 0 1 1 0 0 1 0]	18
5	t6 off	[0 0 1 0 0 1 0 0]	18
6	t3 on	[0 0 1 0 0 1 0 0]	19
6	t3 off	[1 0 1 0 0 0 0 0]	19
7	t1 on	[1 0 1 0 0 0 0 0]	25
7	t1 off	[0 1 1 1 0 0 0 0]	25

Fig. AP6.2.1. *Jurnalul înregistrat de* ***Petri Net Toolbox*** *pe parcursul simulării a 7 evenimente în rețeaua Petri temporizată P considerată în AP6.2.*

2. Pentru a aplica **Teorema 6.6.2**, identificăm circuitele elementare $C_1 = t_1 p_4 t_6 p_6 t_3 p_1$, $C_2 = t_1 p_2 t_2 p_5 t_4 p_7 t_6 p_6 t_3 p_1$ şi $C_3 = t_2 p_5 t_4 p_8 t_5 p_3$, având $D(C_1) = 11$, $M_0(C_1) = 1$, $D(C_2) = 25$, $M_0(C_2) = 1$ şi, respectiv, $D(C_3) = 14$, $M_0(C_3) = 1$. Din relația (BT6.6.3) rezultă că durata unui ciclu de funcționare este egală cu 25 [unități de timp], valoare care a fost obținută şi la punctul 1 (utilizând în mod direct informațiile temporale).

3. Se utilizează opțiunile ***Log File*** şi apoi ***View History***, ambele disponibile sub meniul ***Simulation*** al mediului ***Petri Net Toolbox***, obținându-se rezultatele prezentate în fig. AP6.2.1.

AP6.3.

Ne plasăm în contextul modelului construit în **AP4.1** punctul 1. Ataşăm fiecărei operații o durată de timp, după cum urmează:

- pentru procesorul P_1: crearea unui fişier pe D_1 durează d_{11} = 15 secunde; copierea fişierului pe D_2 durează d_{12} = 10 secunde;
- pentru procesorul P_2: crearea unui fişier pe D_2 durează d_{22} = 3 secunde; copierea fişierului pe D_1 durează d_{21} = 7 secunde.

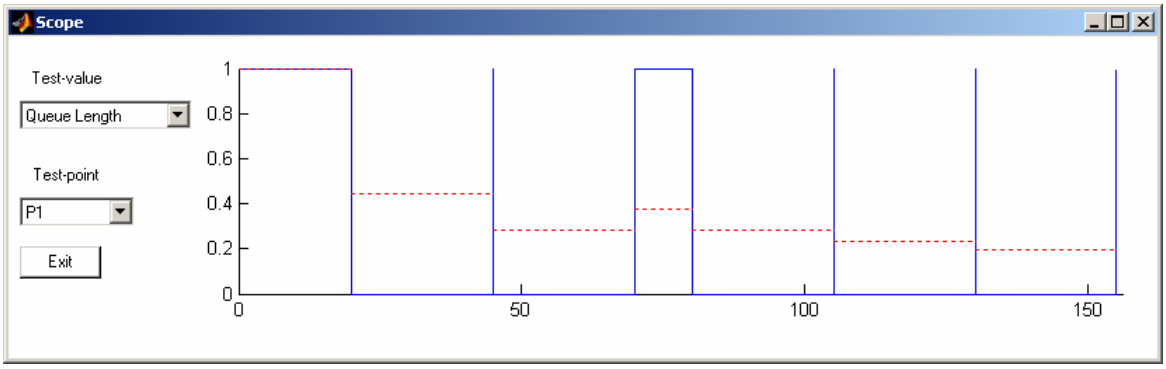
Se consideră că duratele de timp necesare eliberării resurselor sunt neglijabile.

1. Să se adauge pe modelul din fig. AP4.1.2 duratele de timp specifice activităților.
2. Să se utilizeze modelul construit la punctul 1 pentru studierea, cu ajutorul mediului ***Petri Net Toolbox***, a următoarelor situații de funcționare:
   (i) Sistemul funcționează 150 secunde.
      a. Să se traseze câte o diagramă care să reflecte modul de utilizare a fiecăruia din cele două procesoare P_1, P_2.
      b. Să se precizeze numărul de taskuri servite complet din ST_1, respectiv ST_2.
      c. Să se precizeze gradul de utilizare a fiecărui procesor ca valoare a raportului (timpul de utilizare a procesorului / timpul de funcționare a sistemului).
   (ii) Procesorul P_1 serveşte complet exact 1.000 de taskuri.
      a. Să se precizeze cât timp a funcționat sistemul.
      b. Să se precizeze numărul de taskuri servite complet de procesorul P_2.
      c. Să se precizeze gradul de utilizare a fiecărui procesor.
3. Presupunând că se serveşte alternativ câte un task de fiecare tip, să se determine durata unui ciclu de funcționare a sistemului (servirea completă a unui task din ST_1 urmată de servirea completă a unui task din ST_2).
   (i) folosind **Teorema 6.6.1**;
   (ii) folosind direct informațiile temporale privind duratele operațiilor.

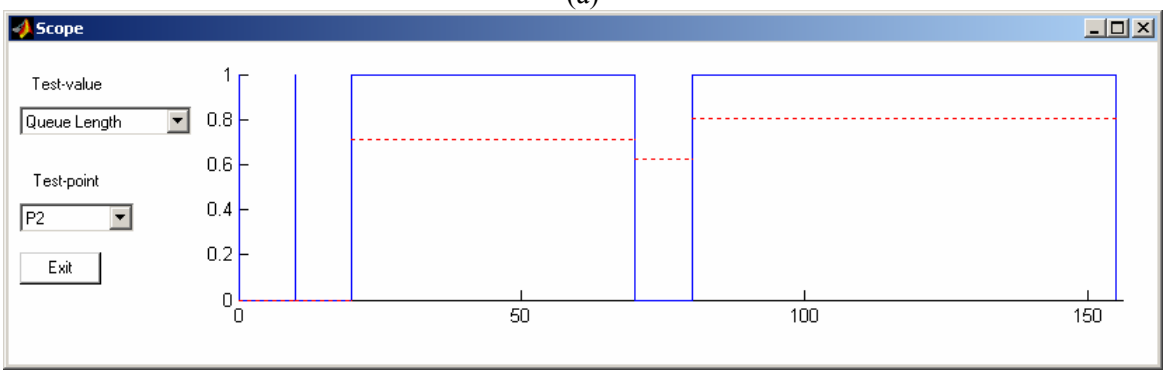
Soluție

1. Modelul netemporizat pentru sistemul de calcul biprocesor, rezultat în urma aplicării algoritmului de sinteză hibridă, este prezentat în fig. AP4.1.2. Duratele de timp precizate în enunț (d_{11}, d_{12}, d_{22} şi d_{21}) se asignează pozițiilor care modelează operațiile (P1D1, P1D1D2, P2D2 şi, respectiv, P2D2D1).

2.(i).a. Utilizând instrumentul ***Scope*** din ***Petri Net Toolbox*** se obțin graficele de evoluție în timp a indicatorilor ***Queue Length*** corespunzători pozițiilor P1 şi P2 prezentate în fig. AP6.3.1. Prezența jetonului în poziția P1 (P2) are semnificația de disponibilitate a procesorului P_1 (respectiv P_2).



(a)



(b)

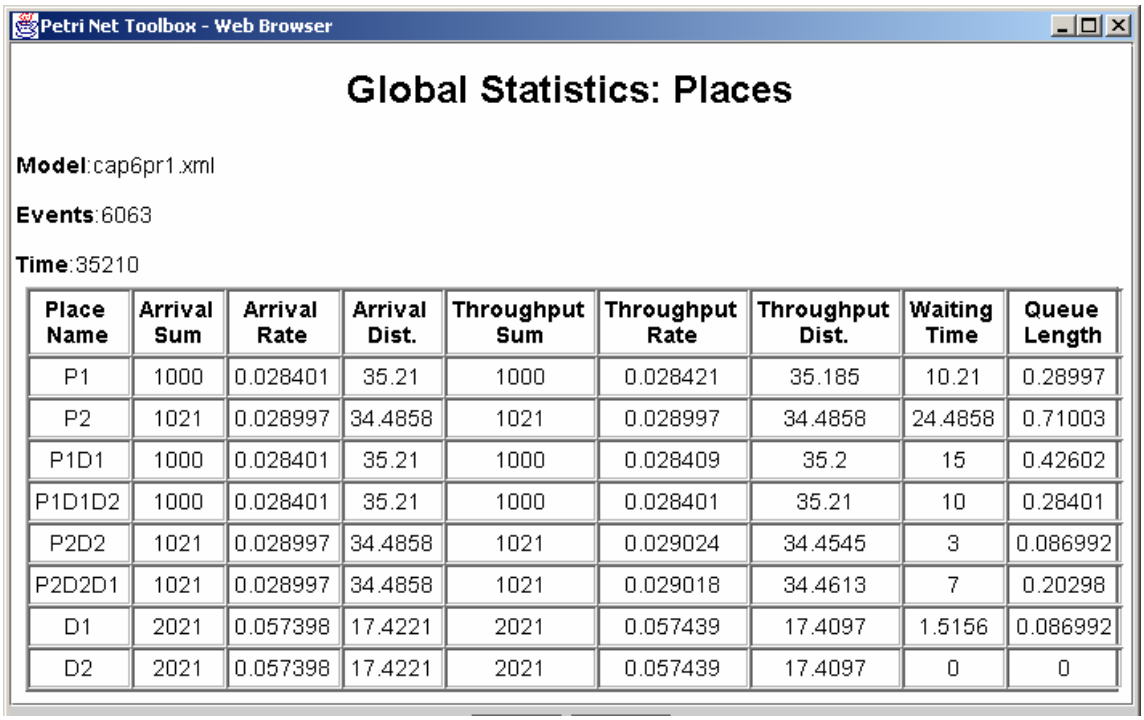
Fig. AP6.3.1. *Evoluția indicatorilor* ***Queue Length*** *corespunzători pozițiilor* (a) *P1 şi* (b) *P2* (*linie continuă – valoare curentă, linie întreruptă – valoare globală rezultată prin mediere pe intervalul de simulare*).

b. Din graficele prezentate în fig. AP6.3.1 se observă că, pe intervalul de simulare de 155 secunde, procesorul P_1 serveşte complet 5 taskuri (fiecare cu durata de 25 secunde) iar procesorul P_2 serveşte complet 3 taskuri (fiecare cu durata de 10 secunde).

c. Gradele de utilizare a celor două procesoare sunt $125/155 = 0.8065$, pentru P_1, respectiv $30/155 = 0.1935$, pentru P_2, după cum rezultă şi din fig. AP6.3.1 prin complementarea față de 1 a valorilor globale pentru indicatorii ***Queue Length***. Aceste valori globale se regăsesc prin utilizarea opțiunii ***Performance / Place Indices***. Deoarece funcționarea sistemului este simulată pe o durată mică, rezultatele obținute nu sunt relevante, după cum se va observa în continuare.

2.(ii).a. Prin intermediul opțiunii ***Tokens Arrived in Place*** din meniul ***Breakpoints*** se poate impune oprirea simulării în regim ***Run Fast*** după ce un număr de 1.000 de jetoane au ajuns în poziția P1 corespunzătoare eliberării procesorului P_1 la terminarea unui task din ST_1. Indicatorii globali de performanță asociați pozițiilor rețelei Petri sunt prezentați în fig. AP6.3.2. Se observă că sistemul a funcționat 35.210 secunde.

b. În intervalul de timp considerat, procesorul P_2 a servit complet 1.021 de taskuri, valoare dată de indicatorul ***Arrival Sum*** pentru poziția P2.


Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:cap6pr1.xml

Events:6063

Time:35210

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
P1	1000	0.028401	35.21	1000	0.028421	35.185	10.21	0.28997
P2	1021	0.028997	34.4858	1021	0.028997	34.4858	24.4858	0.71003
P1D1	1000	0.028401	35.21	1000	0.028409	35.2	15	0.42602
P1D1D2	1000	0.028401	35.21	1000	0.028401	35.21	10	0.28401
P2D2	1021	0.028997	34.4858	1021	0.029024	34.4545	3	0.086992
P2D2D1	1021	0.028997	34.4858	1021	0.029018	34.4613	7	0.20298
D1	2021	0.057398	17.4221	2021	0.057439	17.4097	1.5156	0.086992
D2	2021	0.057398	17.4221	2021	0.057439	17.4097	0	0

Fig. AP6.3.2. *Indicatorii globali de performanță asociați pozițiilor, calculați de* ***Petri Net Toolbox*** *după servirea completă a 1.000 taskuri din* ST_1.

c. Gradele de utilizare a celor două procesoare se obțin prin complementarea față de 1 a valorilor globale pentru indicatorii ***Queue Length*** corespunzători pozițiilor P1 şi P2. Se obțin astfel 71% pentru P_1, respectiv 29% pentru P_2.

3.(i). Aplicăm inegalitatea (BT6.6.2) considerând invarianții P fundamentali determinați la **AP5.2**. Secvența de executări de tranziții care corespunde servirii complete a unui task din ST_1 urmată de servirea completă a unui task din ST_2 este $\sigma = t_1t_2t_3t_4t_5t_6$ cu $\boldsymbol{x} = \bar{\sigma} = [1\ 1\ 1\ 1\ 1\ 1]^T$, pentru care are loc $M_0[\sigma\rangle M_0$. Rezultă astfel $\tau \geq \tau_{\min} = 35$ secunde. Spre deosebire de cazul grafurilor marcate tare conexe (vezi **AP6.1** şi **AP6.2**), **Teorema 6.6.1** nu garantează că valoarea $\tau_{\min}$ este efectiv atinsă de perioada cu care operează rețeaua, deoarece se bazează numai pe o caracterizare algebrică globală a funcționării rețelei. Pentru o analiză mai aprofundată este absolut necesară completarea cu informații provenind din investigarea completă a dinamicii în sensul de la punctul următor.

(ii). Urmărind dinamica rețelei pilotată de secvența σ în regim permanent, constatăm că pentru derularea acestei secvențe sunt necesare, în total, 35 secunde, care se consumă astfel (vezi fig. AP4.1.2): 15 secunde pentru P1D1, 10 secunde pentru P1D1D2, 3 secunde pentru P2D2 şi 7 secunde pentru P2D2D1. Mai mult, observăm că fiecare activitate începe imediat după terminarea celei precedente (spre deosebire de situația când apar fenomene de aşteptare, care va fi ilustrată în **AP6.5**).

AP6.4.

Pentru sistemul de calcul biprocesor cu reprezentarea schematică din fig. AP3.5.1 se doreşte sintetizarea unei structuri de conducere care să asigure funcţionarea corectă a sistemului de calcul în următoarele condiţii:

(c1) Pentru ambele tipuri de taskuri, discul pe care se face copierea este alocat în momentul când s-a încheiat crearea fişierului pe celălalt disc (fără anticipare).
(c2) Taskurile din ST_1 şi ST_2 sunt servite alternativ (câte unul din fiecare tip).
(c3) Servirea începe cu un task din ST_1.

1. Să se reprezinte modelul de tip reţea Petri netemporizată rezultat.
2. Să se adauge pe modelul construit la punctul 1 duratele de timp asociate activităţilor conform specificaţiilor din enunţul **AP6.3**.
3. Presupunând că într-un anumit interval de timp au fost servite complet acelaşi număr de task-uri de tip ST_1 şi ST_2, să se determine gradul de utilizare a fiecărei resurse.

Soluţie

1. Modelul de tip reţea Petri netemporizată al sistemului de calcul împreună cu structura de conducere este prezentat în fig. AP6.4.1.

Poziţiile ST1 şi ST2, nou introduse comparativ cu modelul din fig. AP3.5.2, care satisface condiţia (c1), implementează o structură de control care, suplimentar, asigură şi îndeplinirea condiţiilor (c2) şi (c3) impuse în enunţ.

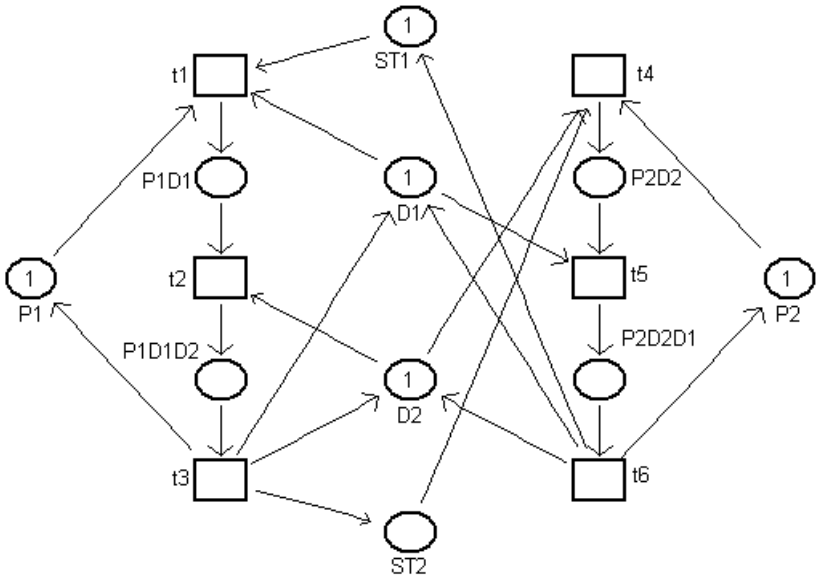
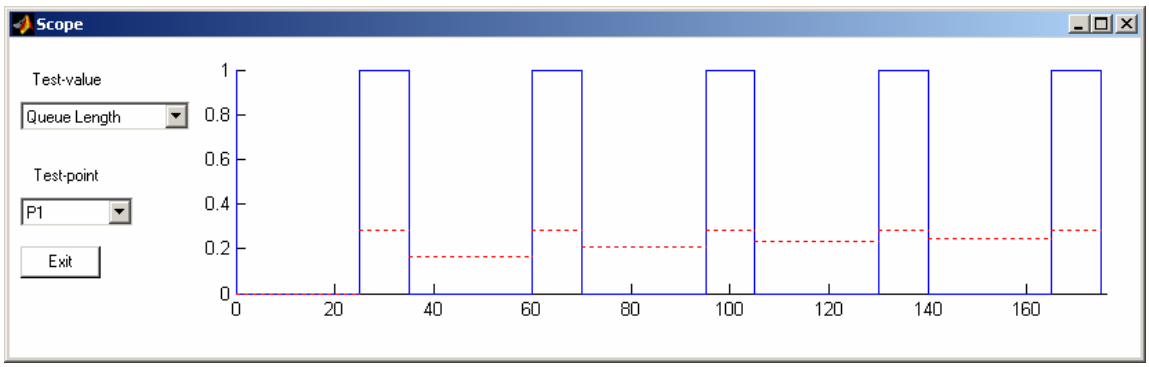
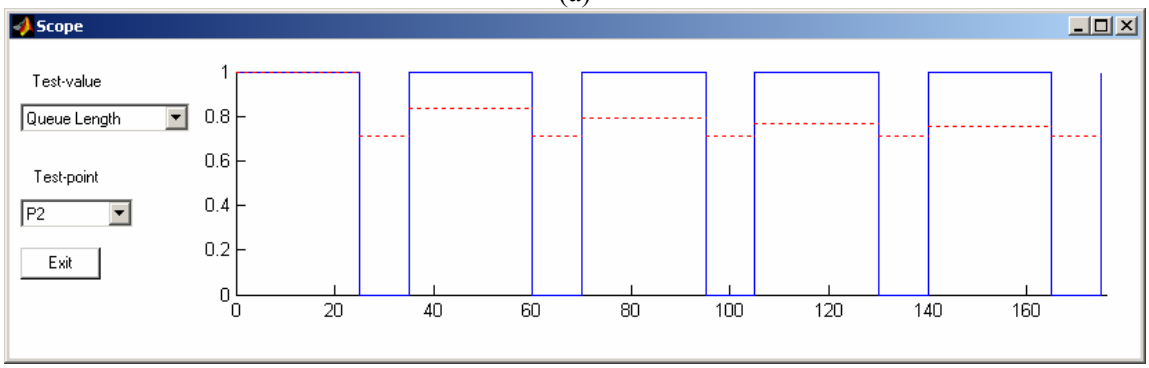


Fig. AP6.4.1. *Modelul de tip reţea Petri utilizat în AP6.4.*

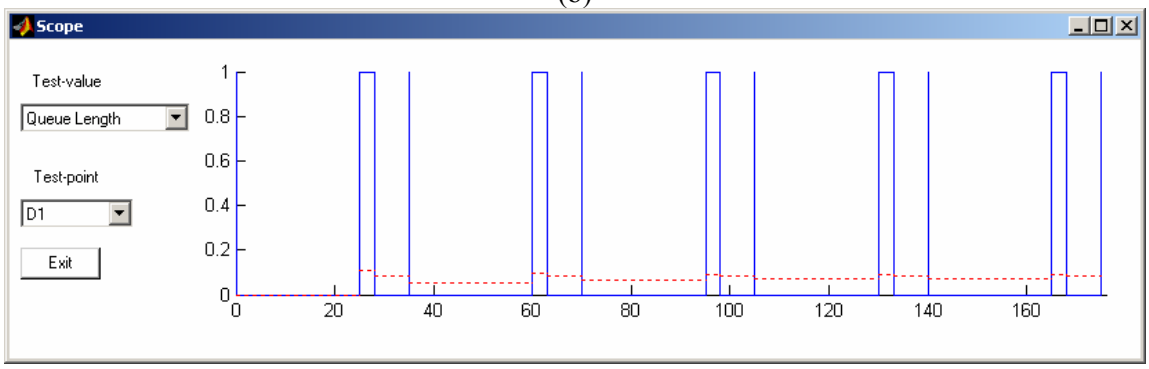
2. Pentru obţinerea modelului temporizat, duratele de timp asociate activităţilor (d_{11}, d_{12}, d_{22} şi d_{21}) se asignează poziţiilor care modelează operaţiile (P1D1, P1D1D2, P2D2 şi, respectiv, P2D2D1).



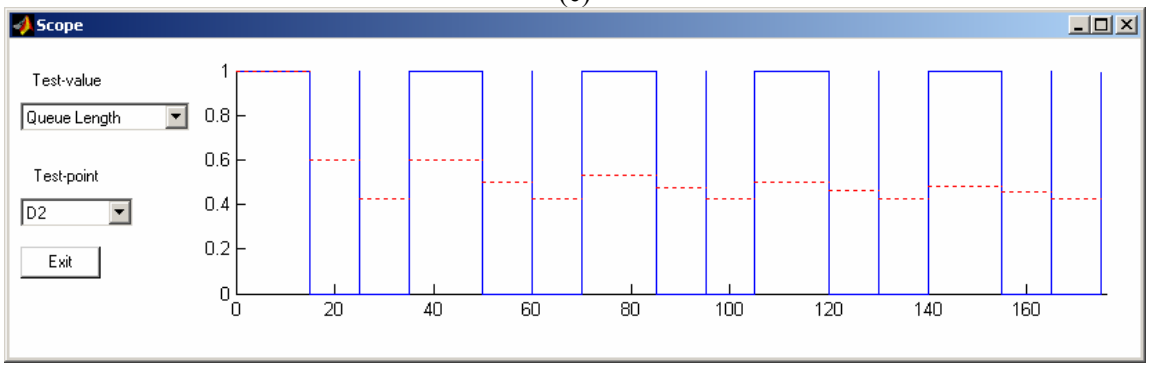
(a)



(b)



(c)



(d)

Fig. AP6.4.2. *Evoluția indicatorilor* ***Queue Length*** *corespunzători pozițiilor* (a) *P1,* (b) *P2,* (c) *D1 şi* (d) *D2* (*linie continuă – valoare curentă, linie întreruptă – valoare globală rezultată prin mediere pe intervalul de simulare*).

3. Utilizând instrumentul ***Scope*** din ***Petri Net Toolbox*** se obțin graficele de evoluție în timp a indicatorilor ***Queue Length*** corespunzători pozițiilor P1, P2, D1 şi D2 prezentate în fig. AP6.4.2. Prezența jetonului în poziția P1 (P2) are semnificația de disponibilitate a procesorului P_1 (respectiv P_2), iar prezența jetonului în poziția D1 (D2) are semnificația de disponibilitate a discului D_1 (respectiv D_2).

Se observă că pe durata a 35 secunde, necesară servirii complete a unui task din ST_1 şi a unui task din ST_2, P_1 este utilizat 15 + 10 = 25 secunde, P_2 este utilizat 3 + 7 = 10 secunde, D_1 este utilizat 15 + 10 + 7 = 32 secunde iar D_2 este utilizat 10 + 3 + 7 = 20 secunde. Gradele de utilizare a resurselor sunt: 71,43% pentru P_1, 28,57% pentru P_2, 91,43% pentru D_1 şi, respectiv, 57,14% pentru D_2.

AP6.5.

Ne plasăm în contextul modelelor construite la **AP4.2** punctul 1, desemnând cu (SC1) modelul de la (i) şi cu (SC2) modelul de la (ii). Duratele activităților desfăşurate în sistemul de fabricație, exprimate în unități de timp, sunt $d_{M1} = 40$, $d_{M2} = 85$ pentru procesarea pe M_1, respectiv M_2, şi $d_{R1} = 20$, $d_{R2} = 20$ pentru descărcarea cu ajutorul lui R a lui M_1, respectiv M_2. Timpul necesitat de eliberarea resurselor M_1, M_2 şi respectiv R este considerat neglijabil. Timpul necesitat de eliberarea unei palete este $d_p = 20$.

1. Să se adauge pe modelele (SC1) şi (SC2) duratele de timp specifice activităților.
2. Să se utilizeze cele două modele pentru analiza în mediul ***Petri Net Toolbox*** a funcționării sistemului pe un interval de timp de 50.000 [unități de timp], precizând:
   (i) durata medie necesară fabricării unui produs finit;
   (ii) numărul de produse finite fabricate;
   (iii) gradul de utilizare a maşinilor M_1 şi M_2.
3. Să se justifice teoretic rezultatele de la 2.(i) folosind **Teorema 6.6.1**.

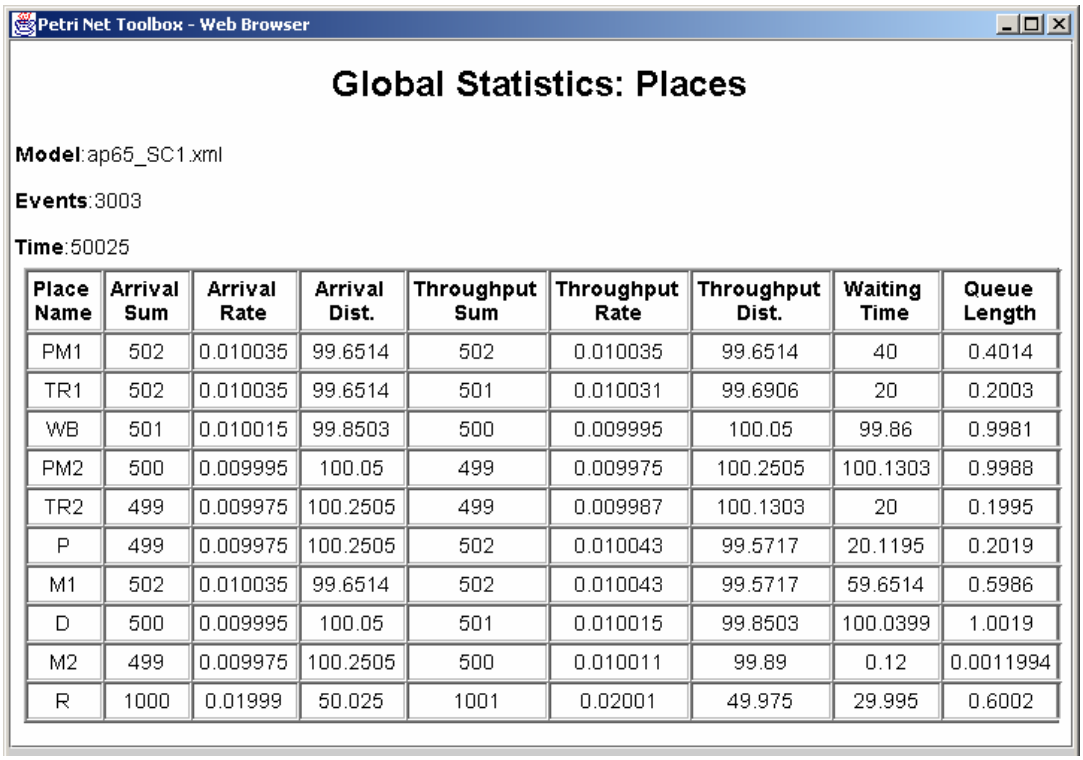
Soluție

1. Modelele de tip rețea Petri netemporizată pentru structurile de conducere (SC1) şi (SC2) sunt prezentate în fig. AP4.2.4 şi, respectiv, AP4.2.5. Pentru obținerea modelelor temporizate, duratele de timp asociate activităților (d_{M1}, d_{M2}, d_{R1} şi d_{R2}) se asignează pozițiilor care modelează operațiile (PM1, PM2, TR1 şi, respectiv, TR2). Durata de eliberare a unei palete, d_p, se asignează poziției care modelează disponibilitatea paletelor (P).

2. Indicatorii globali de performanță referitori la pozițiile rețelelor, determinați în urma simulării modelelor temporizate pe un interval de timp de 50.000 unități de timp, sunt prezentați în fig. AP6.5.1 (pentru SC1) şi fig. AP6.5.2 (pentru SC2). Pe baza informațiilor din fig. AP6.5.1 şi fig. AP6.5.2 se pot evalua performanțele sistemului de fabricație:

(i) durata medie necesară fabricării unui produs finit:
- pentru SC1: 100,25 [unități de timp],
- pentru SC2: 85,24 [unități de timp],

valorile fiind date de indicatorul ***Arrival Distance*** pentru poziția P;


Petri Net Toolbox - Web Browser

Global Statistics: Places

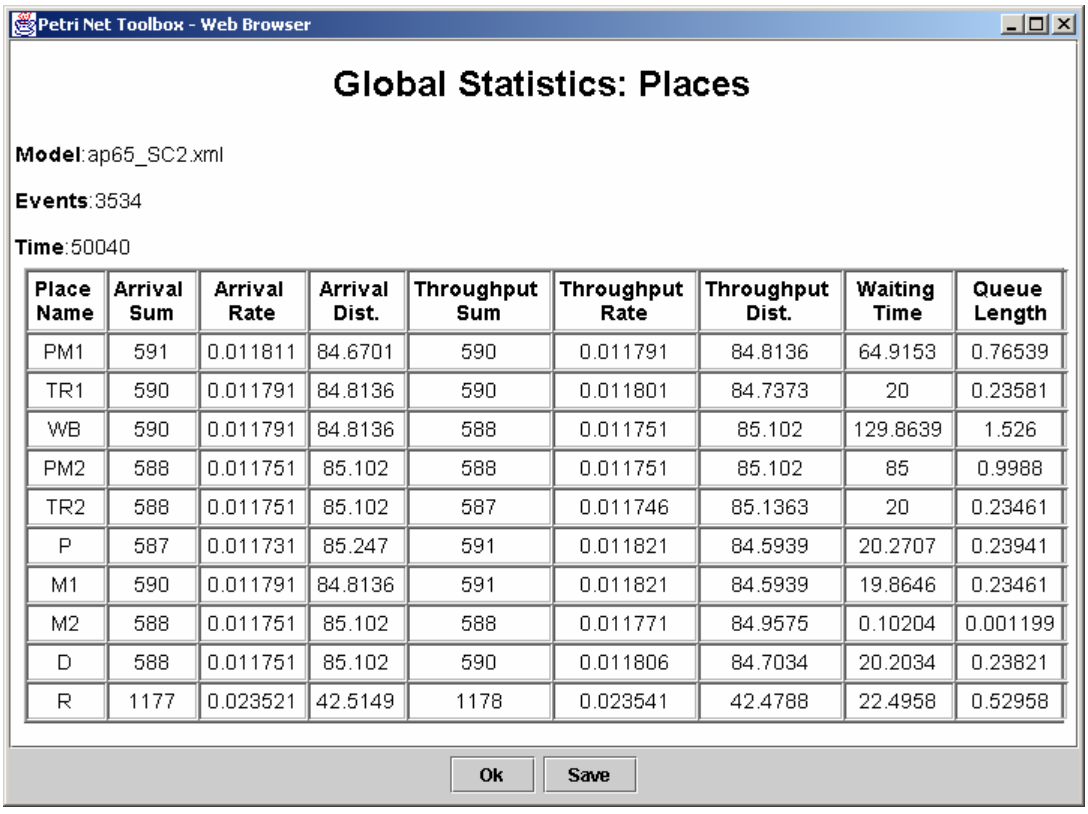
Model:ap65_SC1.xml

Events:3003

Time:50025

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
PM1	502	0.010035	99.6514	502	0.010035	99.6514	40	0.4014
TR1	502	0.010035	99.6514	501	0.010031	99.6906	20	0.2003
WB	501	0.010015	99.8503	500	0.009995	100.05	99.86	0.9981
PM2	500	0.009995	100.05	499	0.009975	100.2505	100.1303	0.9988
TR2	499	0.009975	100.2505	499	0.009987	100.1303	20	0.1995
P	499	0.009975	100.2505	502	0.010043	99.5717	20.1195	0.2019
M1	502	0.010035	99.6514	502	0.010043	99.5717	59.6514	0.5986
D	500	0.009995	100.05	501	0.010015	99.8503	100.0399	1.0019
M2	499	0.009975	100.2505	500	0.010011	99.89	0.12	0.0011994
R	1000	0.01999	50.025	1001	0.02001	49.975	29.995	0.6002

Fig. AP6.5.1. *Indicatorii globali de performanță referitori la pozițiile din modelul* (*SC1*).


Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap65_SC2.xml

Events:3534

Time:50040

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
PM1	591	0.011811	84.6701	590	0.011791	84.8136	64.9153	0.76539
TR1	590	0.011791	84.8136	590	0.011801	84.7373	20	0.23581
WB	590	0.011791	84.8136	588	0.011751	85.102	129.8639	1.526
PM2	588	0.011751	85.102	588	0.011751	85.102	85	0.9988
TR2	588	0.011751	85.102	587	0.011746	85.1363	20	0.23461
P	587	0.011731	85.247	591	0.011821	84.5939	20.2707	0.23941
M1	590	0.011791	84.8136	591	0.011821	84.5939	19.8646	0.23461
M2	588	0.011751	85.102	588	0.011771	84.9575	0.10204	0.001199
D	588	0.011751	85.102	590	0.011806	84.7034	20.2034	0.23821
R	1177	0.023521	42.5149	1178	0.023541	42.4788	22.4958	0.52958

Ok Save

Fig. AP6.5.2. *Indicatorii globali de performanță referitori la pozițiile din modelul* (*SC2*).

(ii) numărul de produse finite fabricate:
- pentru SC1: 499 piese,
- pentru SC2: 587 piese,

valorile fiind date de indicatorul ***Arrival Sum*** pentru poziţia P;

(iii) gradul de utilizare a maşinilor M_1 şi M_2:
- pentru SC1: 40,1% pentru M_1 şi 99,8% pentru M_2,
- pentru SC2: 76,5% pentru M_1 şi 99,8% pentru M_2,

valorile fiind date de indicatorul ***Queue Length*** pentru poziţiile PM1 şi PM2.

Observaţie: Dacă, pentru modelul (SC1), se compară indicatorul ***Waiting Time*** pentru poziţia PM2 din tabelul prezentat în fig. AP6.5.1 (100,13 [unităţi de timp]) cu valoarea cunoscută a timpului de procesare a unei piese pe maşina M_2 (85 [unităţi de timp]) se constată că maşina M_2 este ocupată cu piesă şi după încheierea prelucrării, piesa aşteptând disponibilitatea robotului pentru descărcare. Această situaţie conduce la interpretarea că valoarea de 99,8% menţionată mai sus pentru M_2 măsoară ceea ce am putea numi drept „grad de ocupare" al resursei, gradul de utilizare propriu-zisă fiind mai mic. O remarcă similară se poate face şi în cazul modelului (SC2) cu referire la maşina M_1.

3. Pentru modelul (SC1) aplicăm inegalitatea (BT6.6.2) cu invarianţii *P* fundamentali determinaţi la **AP5.3**. Secvenţa de executări de tranziţii care corespunde prelucrării complete a unei piese în sistemul de fabricaţie este $\sigma = t_1 t_2 t_3 t_4 t_5 t_6$ cu $\boldsymbol{x} = \overline{\sigma} = [1\ 1\ 1\ 1\ 1\ 1]^T$, pentru care are loc $M_0[\sigma\rangle M_0$. Rezultă astfel că durata minimă a unui ciclu de funcţionare a sistemului este $\tau_{\min} = 85$ [unităţi de timp]. Valoarea obţinută prin simulare, şi anume 100 [unităţi de timp], este strict mai mare decât $\tau_{\min}$. Aceasta se datorează faptului că la terminarea prelucrării unei piese pe maşina M_2 robotul nu incepe imediat descărcarea acesteia, fiind ocupat cu transportarea unei alte piese de pe M_1 în depozit (a se vedea deosebirea faţă de situaţia ilustrată în **AP6.3**).

AP6.6.

Fie sistemul de fabricaţie cu modelul (SC1) din **AP6.5**.

1. Se consideră că durata constantă necesară eliberării paletelor d_P este ajustabilă între limitele $[d_{P\min}, d_{P\max}]$, unde $d_{P\min} = 20$ şi $d_{P\max} = 30$.
   (i) Folosind meniul ***Design*** al mediului ***Petri Net Toolbox*** să se realizeze o reprezentare grafică a duratei medii necesare fabricării unui produs finit în funcţie de valoarea d_P, pentru un timp de funcţionare de 50.000 [unităţi de timp].
   (ii) Să se precizeze în ce condiţii de alegere a valorii d_P durata medie de fabricare a unui produs finit este minimă.
   (iii) Se consideră valoarea d_P obţinută la punctul (ii). Să se precizeze dacă prin ajustarea duratelor de timp ale activităţilor de transport, se poate obţine o durată medie de fabricare a unui produs inferioară celei de la (ii) şi să se motiveze.

2. Pentru d_P din **AP6.5** se consideră că duratele celor două operații de transport realizate de robot, d_{R1}, d_{R2}, sunt ajustabile independent între limitele $[d_{R1min}, d_{R1max}]$ și, respectiv, $[d_{R2min}, d_{R2max}]$, unde $d_{R1min} = d_{R2min} = 18$ și $d_{R1max} = d_{R2max} = 22$.
   (i) Folosind meniul ***Design*** al mediului ***Petri Net Toolbox*** să se realizeze o reprezentare grafică a duratei medii necesare fabricării unui produs finit în funcție de valorile d_{R1} și d_{R2}, considerând loturi de câte 1.000 produse finite.
   (ii) Să se precizeze în ce condiții de alegere a valorilor d_{R1} și d_{R2} durata medie de fabricare a unui produs finit este minimă.
   (iii) Se consideră valorile d_{R1} și d_{R2} obținute la punctul (ii). Să se precizeze dacă prin ajustarea duratei de timp a eliberării paletelor se poate obține o durată medie de fabricare a unui produs inferioară celei de la (ii) și să se motiveze.

Soluție

1.(i). Utilizând instrumentul ***Design*** disponibil în mediul ***Petri Net Toolbox***, se obține reprezentare grafică din fig. AP6.6.1 a duratei medii necesare fabricării unui produs finit (valoarea indicatorului ***Arrival Distance*** pentru poziția P) în funcție de valoarea d_P, considerată drept parametru, pentru un timp de funcționare de 50.000 [unități de timp].

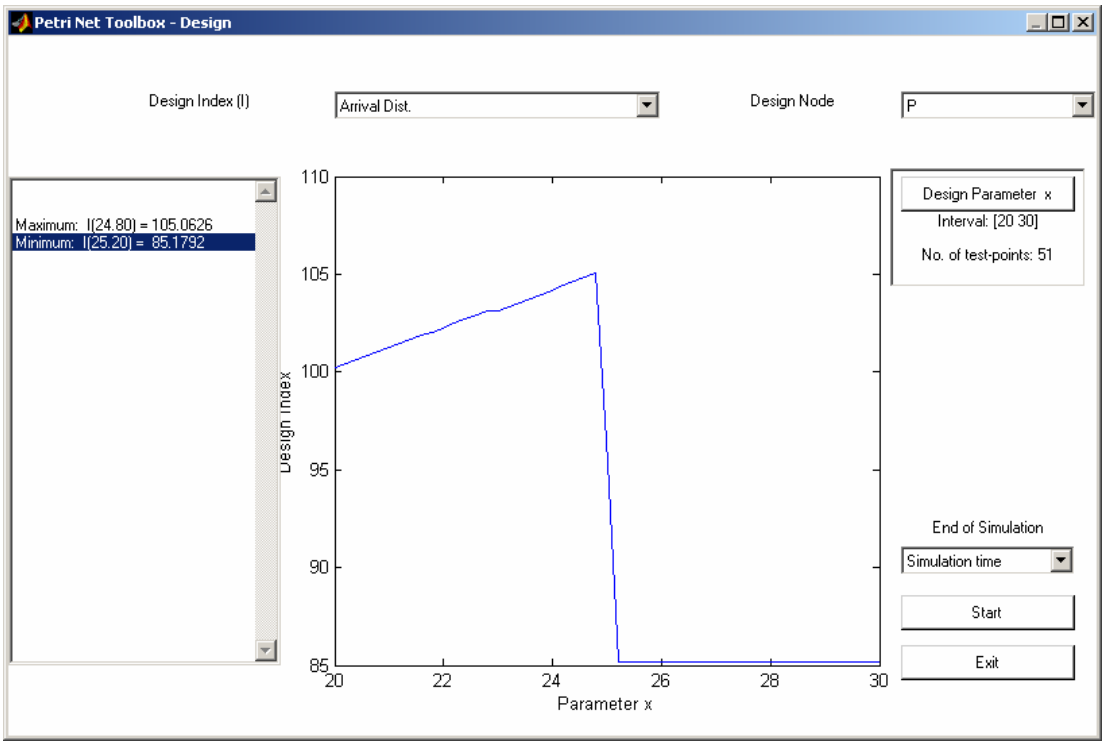


Fig. AP6.6.1 *Evoluția duratei medii de fabricare a unei piese în funcție de durata de eliberare a unei palete.*

(ii). Consultând graficul din fig. AP6.6.1 se observă că pentru o durată de eliberare a paletelor strict mai mare de 25 [unități de timp], durata medie de fabricare este minimă și are valoarea 85,18 [unități de timp]. Cititorului îi este recomandat să interpreteze alura graficului din

fig. AP6.6.1 şi să justifice dependența duratei medii de fabricare de durata de eliberare a paletelor.

(iii). Maşina M_2 (ce poate prelucra o singură piesă la un moment dat) reprezintă resursa de tip „bottleneck" din sistemul de fabricație, pe ea desfăşurându-se operația cu cea mai mare durată. Deoarece operațiile efectuate asupra unei piese se desfăşoară secvențial, durata medie a unui ciclu de fabricație nu poate fi redusă sub durata operației care se desfăşoară pe resursa „bottleneck", adică 85 [unități de timp], oricare ar fi duratele de timp ale activităților de transport. Valoarea de 85,18 [unități de timp] (determinată de ***Petri Net Toolbox***) este afectată de o eroare sub 0,25% datorate simulării numerice.

2.(i). Utilizând instrumentul ***Design*** disponibil în mediul ***Petri Net Toolbox***, se obține reprezentare grafică din fig. AP6.6.2 a duratei medii necesare fabricării unui produs finit (valoarea indicatorului ***Arrival Distance*** pentru poziția P) în funcție de valorile d_{R1} şi d_{R2}, considerate drept parametri, pentru loturi de câte 1.000 produse finite.

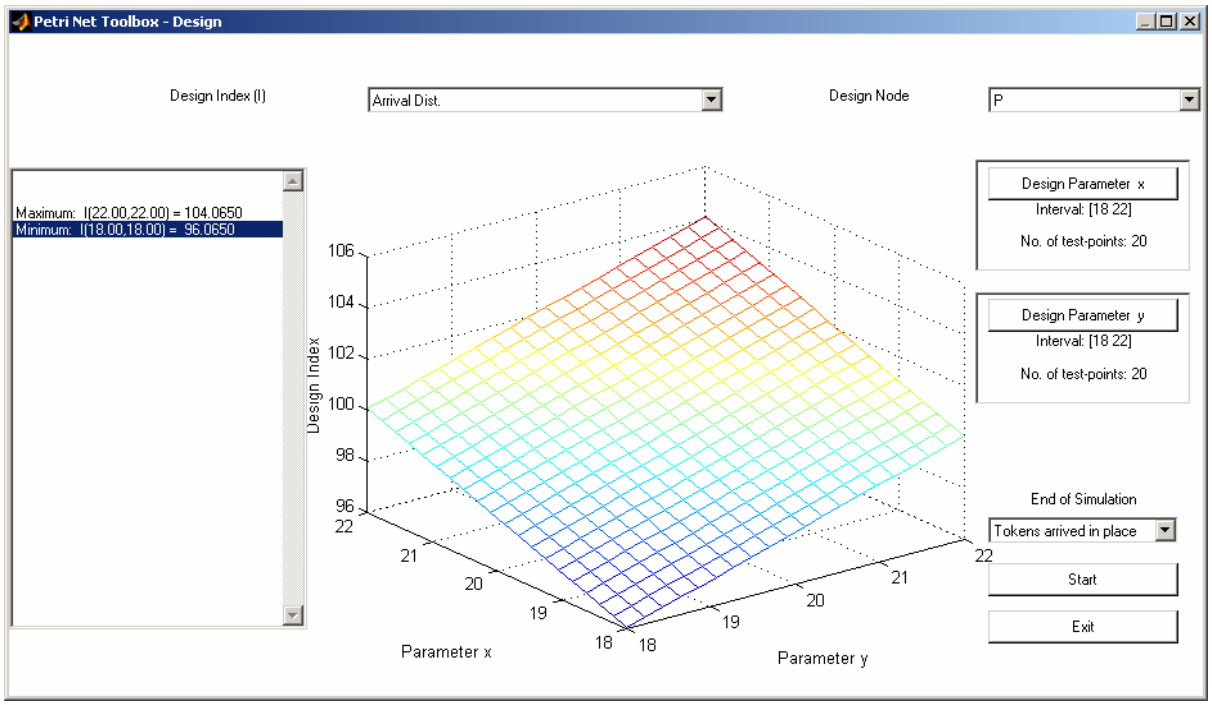


Fig. AP6.6.2 *Evoluția duratei medii de fabricare a unei piese în funcție de duratele activităților de transport.*

(ii). Consultând graficul din fig. AP6.6.2 se observă că pentru duratele activităților de transport $d_{R1} = d_{R2} = 18$ [unități de timp] durata unui ciclu de fabricație este minimă şi are valoarea 96,06 [unități de timp]. Cititorului îi este recomandat să interpreteze alura suprafeței din fig. AP6.6.2 şi să justifice dependența duratei ciclului de fabricație de duratele activităților de transport.

(iii). Conform observațiilor de la punctul 1.(ii) ajustând durata de eliberare a paletelor se poate obține o durată a ciclului de fabricație apropiată de 85 [unități de timp].

AP6.7.

Fie sistemul de fabricație cu modelul (SC1) din **AP6.5** cu singura deosebire că pentru transportul pieselor se folosesc 4 palete în loc de 3. Pentru obținerea aceluiași produs finit se utilizează două loturi de piese brute diferite L_1, L_2. Pentru piesele din L_1, duratele operațiilor sunt cele din **AP6.5**. Piesele din L_2 diferă de cele din L_1 numai prin durata procesării pe mașina M_2, notată $d^*_{M2} = 75$ [unități de timp].

1. Să se simuleze funcționarea sistemului de fabricație pentru un interval de 50.000 [unități de timp] considerând piese din lotul L_1, și apoi din lotul L_2 și să se precizeze durata medie de fabricație a unui produs finit. Să se comenteze și explice rezultatele prin prisma structurii de conducere (SC1).
2. Sistemul funcționează alimentat cu piese din lotul L_2. Să se traseze câte o diagramă care să reflecte modul de utilizare a resurselor M_1, M_2, R, pentru un interval de timp de 250 [unități de timp].

Soluție

1. Pentru piesele din lotul L_1 sistemul se blochează (ajunge în deadlock), în timp ce pentru piesele din lotul L_2 sistemul funcționează fără blocaj. Detaliind, pentru piesele din lotul L_1, având durata de prelucrare pe mașina M_2 de 85 [unități de timp], poate apărea situația în care robotul disponibil este alocat pentru descărcarea lui M_1 deși depozitul este plin și există piesă în curs de prelucrare pe M_2. O atare situație nu poate apărea în cazul prelucrării pieselor din lotul L_2 (cititorul va justifica de ce).

Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap6_7_L2.xml

Events:4002

Time:50015

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
PM1	669	0.013376	74.7608	668	0.013356	74.8728	54.9626	0.73408
TR1	668	0.013356	74.8728	668	0.013365	74.8204	20	0.26712
WB	668	0.013356	74.8728	666	0.013316	75.0976	109.8949	1.4634
PM2	666	0.013316	75.0976	666	0.013316	75.0976	75.0075	0.9988
TR2	666	0.013316	75.0976	665	0.013311	75.1278	20	0.26592
P	665	0.013296	75.2105	669	0.013385	74.7085	20.2392	0.27072
M1	668	0.013356	74.8728	669	0.013385	74.7085	19.8804	0.26592
D	666	0.013316	75.0976	668	0.013365	74.8204	40.1796	0.53664
M2	666	0.013316	75.0976	666	0.013336	74.985	0.09009	0.0011996
R	1333	0.026652	37.5206	1334	0.026672	37.4925	17.5075	0.46696

Fig. AP6.7.1. *Indicatorii globali de performanță pentru pozițiile modelului* (*SC1*) *corespunzător prelucrării pieselor din lotul* L_2.

Pentru piesele din lotul L_2 durata medie de fabricație a unui produs finit, anume 75,21 [unități de timp] reprezintă valoarea indicatorului ***Arrival Distance*** pentru poziția P (fig. AP6.7.1).

2. Considerând sistemul de fabricație alimentat cu piese din lotul L_2, diagramele din fig. AP6.7.2 (obținute cu ajutorul instrumentului ***Scope*** din ***Petri Net Toolbox***) prezintă evoluția indicatorilor ***Queue Length*** corespunzători pozițiilor PM1, PM2 şi R pe un interval de circa 250 [unități de timp]. Prezența jetonului în poziția PM1 (PM2) are semnificația de utilizare a mașinii M_1 (respectiv M_2), iar prezența jetonului în poziția R are semnificația de disponibilitate a robotului R.

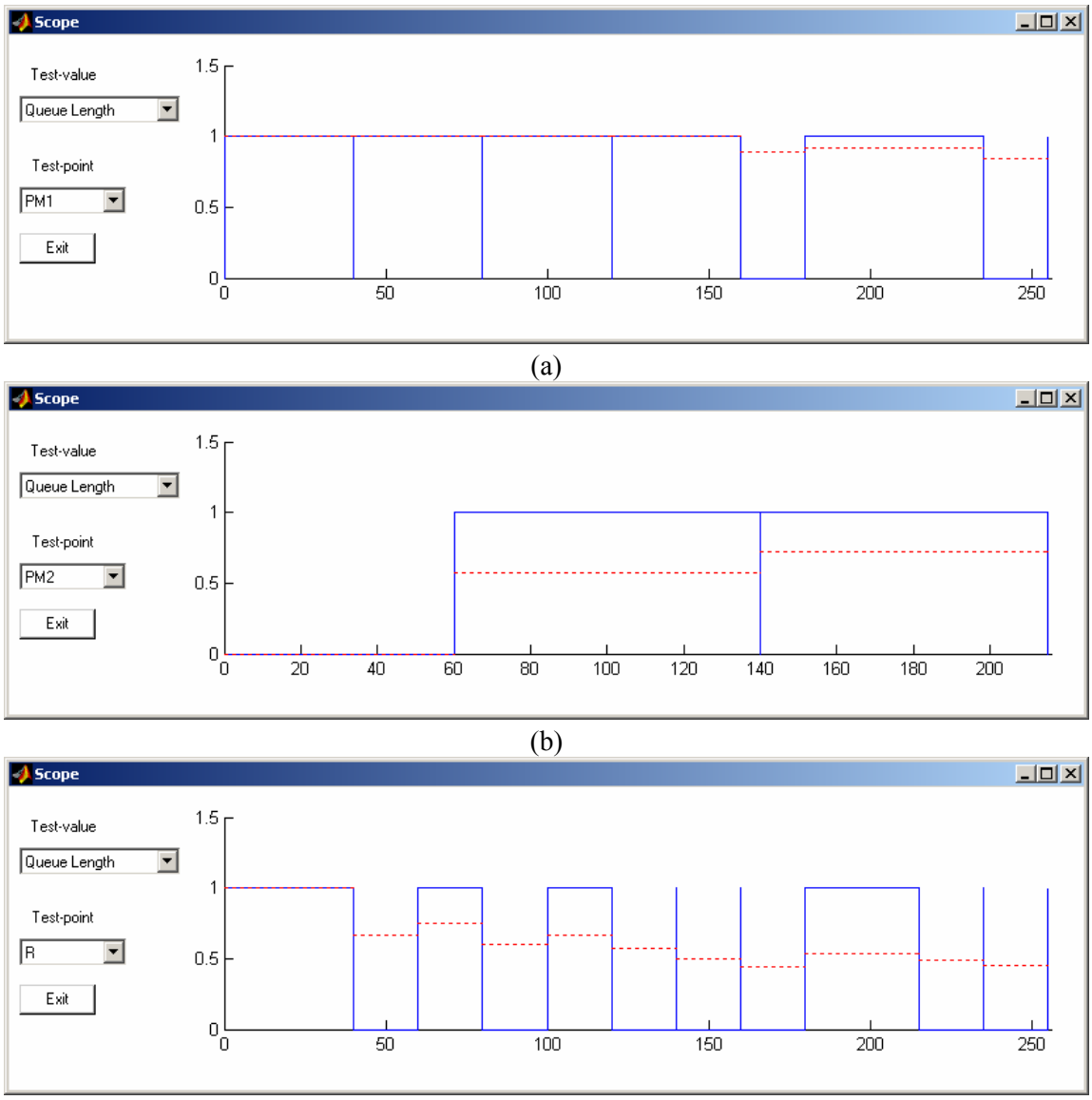


Fig. AP6.7.2. *Evoluția indicatorilor* ***Queue Length*** *corespunzători pozițiilor* (a) *PM1,* (b) *PM2 şi* (c) *R*
(*linie continuă – valoare curentă, linie întreruptă – valoare globală rezultată prin mediere pe intervalul de simulare*).

AP6.8.

Se consideră o celulă de fabricație flexibilă orientată pe producția diversificată (eng. *job shop*). Celula conține două mașini de prelucrare universale M_1, M_2 și sisteme de transport adecvate. Se fabrică 3 tipuri de produse P_1, P_2, P_3 care necesită următoarele secvențe de prelucrări:

- pentru produsul P_1: prelucrare 1 pe M_1 (durata $d_{11} = 3$ min), prelucrare 2 pe M_2 (durata $d_{12} = 5$ min);
- pentru produsul P_2: prelucrare 1 pe M_2 (durata $d_{22} = 6$ min), prelucrare 2 pe M_1 (durata $d_{21} = 4$ min);
- pentru produsul P_3: prelucrare 1 pe M_1 (durata $d_{31} = 7$ min), prelucrare 2 pe M_2 (durata $d_{32} = 3$ min).

Duratele pentru operațiile de transport și pentru eliberarea resurselor sunt considerate neglijabile. Se presupune că numărul de piese brute este suficient de mare.

Se consideră următoarele două strategii de planificare a servirii produselor (clienților) P_1, P_2, P_3 de către resursele M_1, M_2:

(S_1) M_1: P_1, P_2, P_3 ; M_2: P_1, P_2, P_3;

(S_2) M_1: P_1, P_3, P_2 ; M_2: P_1, P_3, P_2.

1. Să se construiască modelul tip rețea Petri temporizată *P* al căilor de operații pentru P_1, P_2, P_3.
2. Să se atașeze modelului construit la punctul 1 structurile de control care asigură planificarea servirii conform (S_1) și, respectiv, (S_2).
3. Să se utilizeze aceste modele în mediul ***Petri Net Toolbox*** pentru a determina duratele medii de producere a celor trei tipuri de produse, dacă funcționarea celulei durează 100 ore.
4. Să se justifice teoretic rezultatele de la punctul 3 folosind **Teorema 6.6.2** pentru a determina durata minimă a unui ciclu de funcționare.

Soluție

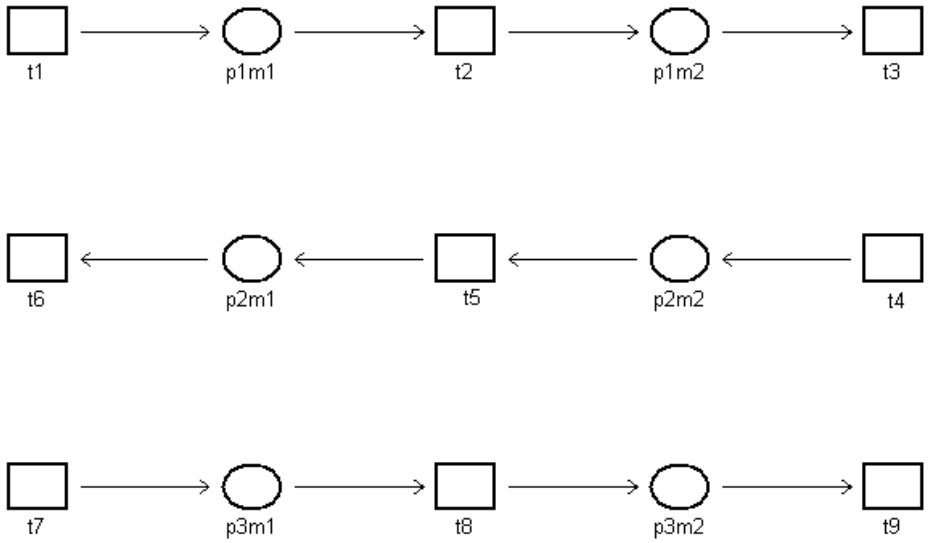


Fig. AP6.8.1. *Modelul de tip rețea Petri al căilor de operații din celula de fabricație studiată în AP6.8.*

1. Modelul de tip reţea Petri al căilor de operaţii pentru cele trei tipuri de piese este prezentat în fig. AP6.8.1. Operaţia de prelucrare a unei piese de tipul P_i, $i = 1,2,3$, pe maşina M_j, $j = 1,2$, este reprezentată prin poziţia notată **p*i*m*j***, căreia îi este asignată durata d_{ij}.

2. Prin adăugarea structurilor de control care asigură planificarea servirii conform celor două strategii rezultă reţelele Petri prezentate în fig. AP6.8.2 (pentru S_1) şi fig. AP6.8.3 (pentru S_2). Poziţiile notate p7, p8, ..., p12, care aparţin structurii de control, nu sunt temporizate.

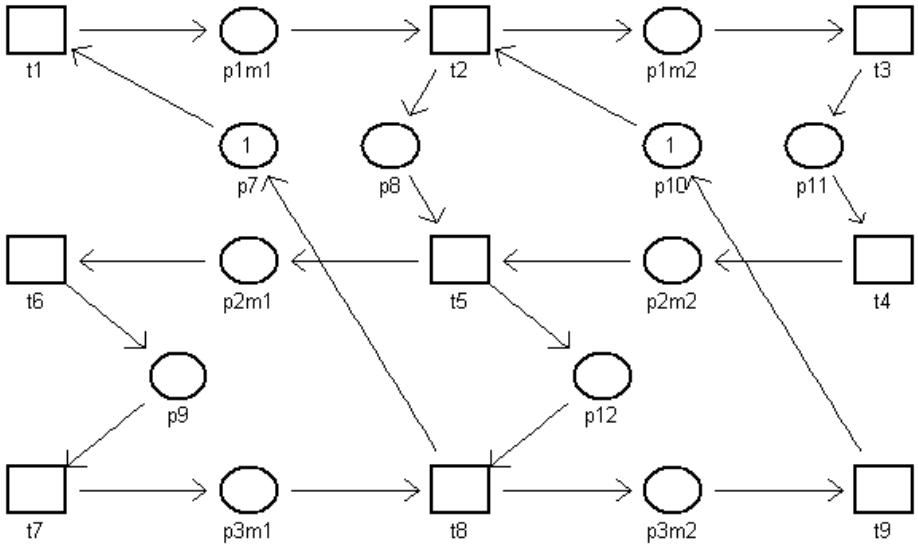


Fig. AP6.8.2. *Modelul de tip reţea Petri pentru celula de fabricaţie cu structura de control care asigură planificarea servirii clienţilor conform strategiei* (S_1).

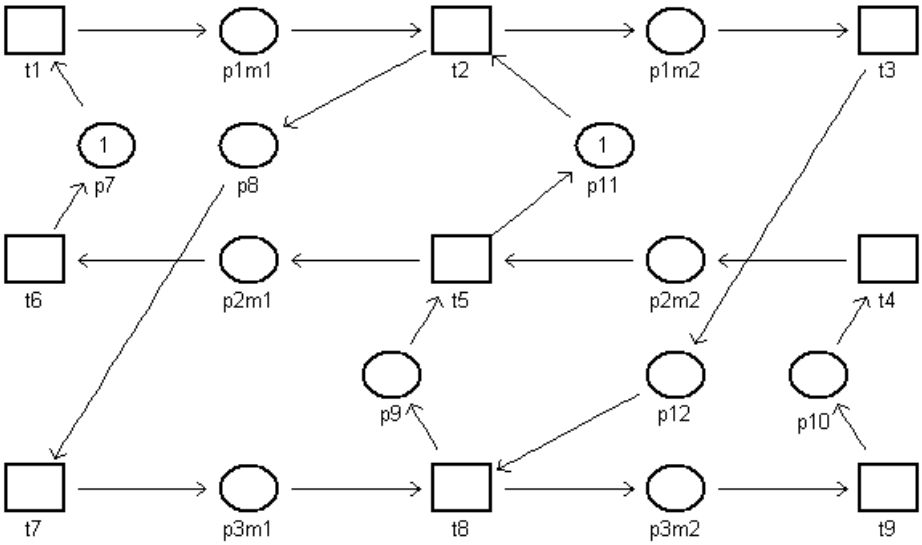
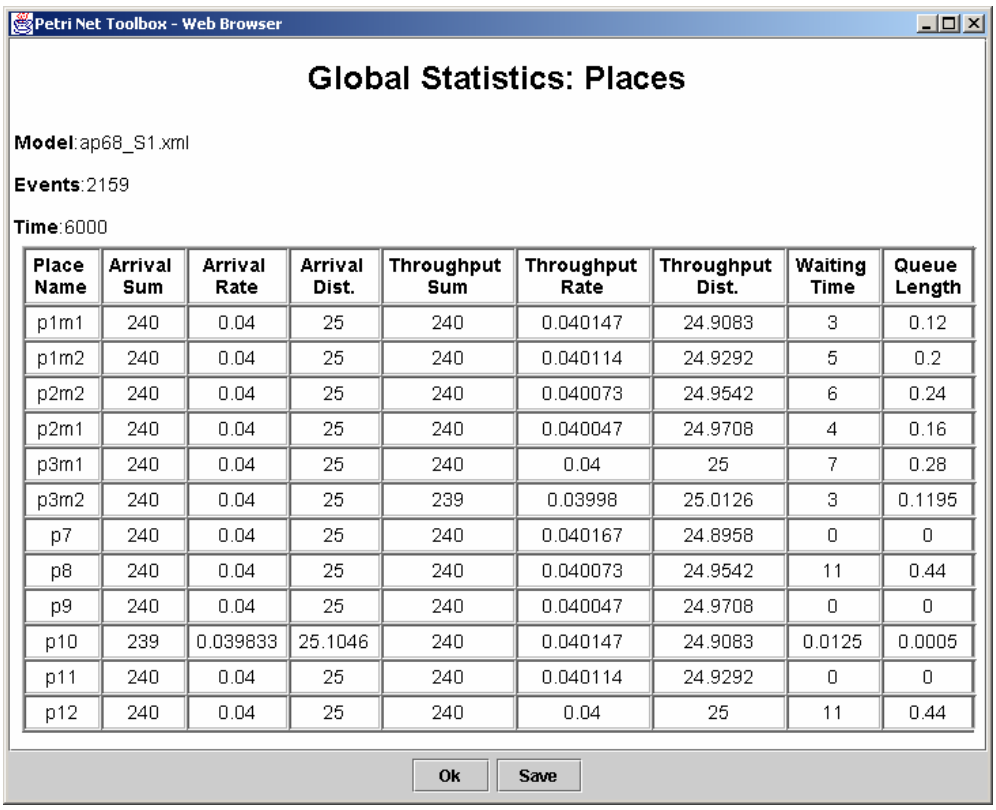


Fig. AP6.8.3. *Modelul de tip reţea Petri pentru celula de fabricaţie cu structura de control care asigură planificarea servirii clienţilor conform strategiei* (S_2).


Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap68_S1.xml

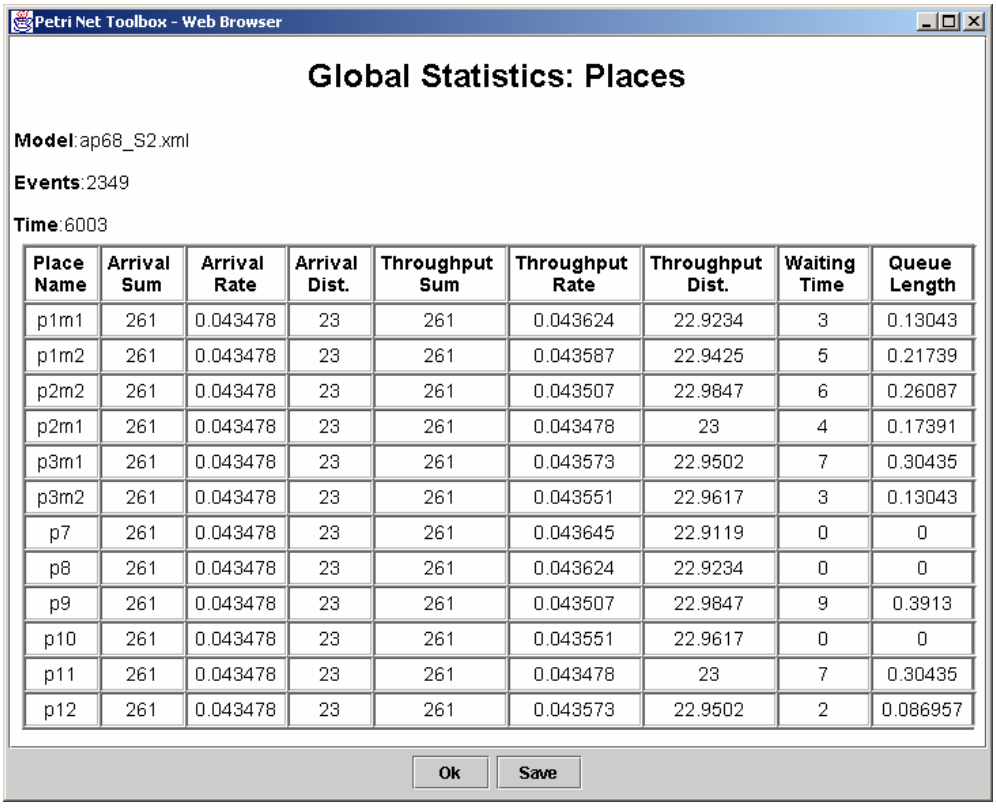
Events:2159

Time:6000

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
p1m1	240	0.04	25	240	0.040147	24.9083	3	0.12
p1m2	240	0.04	25	240	0.040114	24.9292	5	0.2
p2m2	240	0.04	25	240	0.040073	24.9542	6	0.24
p2m1	240	0.04	25	240	0.040047	24.9708	4	0.16
p3m1	240	0.04	25	240	0.04	25	7	0.28
p3m2	240	0.04	25	239	0.03998	25.0126	3	0.1195
p7	240	0.04	25	240	0.040167	24.8958	0	0
p8	240	0.04	25	240	0.040073	24.9542	11	0.44
p9	240	0.04	25	240	0.040047	24.9708	0	0
p10	239	0.039833	25.1046	240	0.040147	24.9083	0.0125	0.0005
p11	240	0.04	25	240	0.040114	24.9292	0	0
p12	240	0.04	25	240	0.04	25	11	0.44

Ok Save

Fig. AP6.8.4. *Indicatorii globali de performanță pentru pozițiile rețelei din fig. AP6.8.2.*


Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap68_S2.xml

Events:2349

Time:6003

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
p1m1	261	0.043478	23	261	0.043624	22.9234	3	0.13043
p1m2	261	0.043478	23	261	0.043587	22.9425	5	0.21739
p2m2	261	0.043478	23	261	0.043507	22.9847	6	0.26087
p2m1	261	0.043478	23	261	0.043478	23	4	0.17391
p3m1	261	0.043478	23	261	0.043573	22.9502	7	0.30435
p3m2	261	0.043478	23	261	0.043551	22.9617	3	0.13043
p7	261	0.043478	23	261	0.043645	22.9119	0	0
p8	261	0.043478	23	261	0.043624	22.9234	0	0
p9	261	0.043478	23	261	0.043507	22.9847	9	0.3913
p10	261	0.043478	23	261	0.043551	22.9617	0	0
p11	261	0.043478	23	261	0.043478	23	7	0.30435
p12	261	0.043478	23	261	0.043573	22.9502	2	0.086957

Ok Save

Fig. AP6.8.5. *Indicatorii globali de performanță pentru pozițiile rețelei din fig. AP6.8.3.*

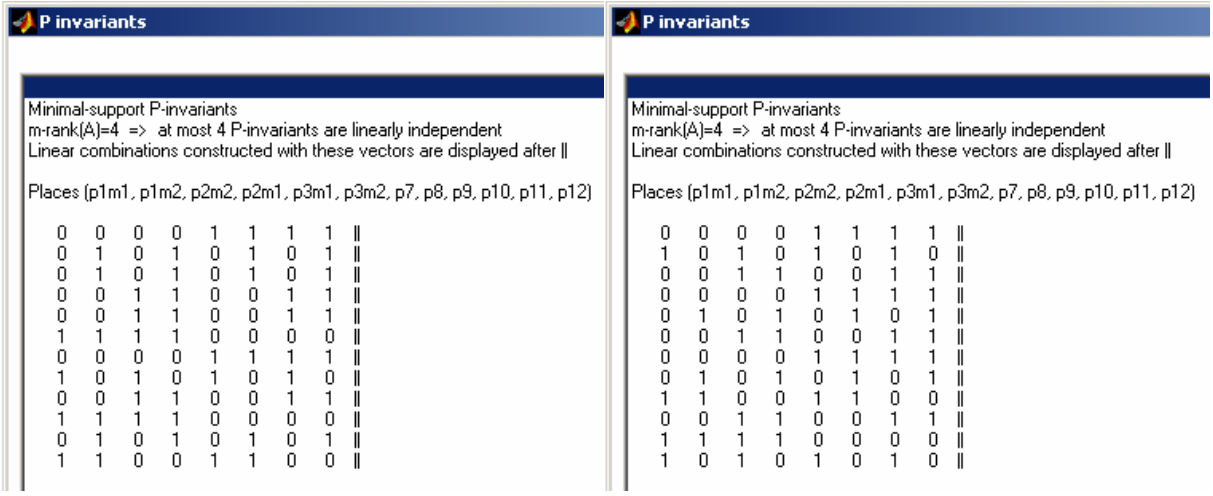
3. Rezultatele furnizate de ***Petri Net Toolbox*** după simularea funcţionării celulei de fabricaţie timp de 100 ore sunt prezentate în fig. AP6.8.4 pentru S_1 şi fig. AP6.8.5 pentru S_2. Valorile indicatorilor ***Throughput Distance*** pentru poziţiile p1m2, p2m1 şi p3m2 reprezintă duratele medii de fabricare a unei piese şi anume: circa 25 minute dacă se aplică strategia (S_1) şi circa 23 minute dacă se aplică strategia (S_2).

4. Se observă că reţeaua Petri din fig. AP6.8.2 este graf marcat tare conex, astfel încât aplicarea **Teoremei 6.6.2** are obiect. Pentru aceasta determinăm mai întâi circuitele elementare utilizând ***Petri Net Toolbox*** cu opţiunea ***P-Invariants*** din meniul ***Properties / Invariants*** care permit identificarea poziţiilor din fiecare circuit. Drept urmare, invarianţilor din fig. AP6.8.3.(a), parcurşi de la stânga la dreapta, le corespund următoarele circuite:

C_1 = t2, p8, t5, p12, t8, p3m2, t9, p10;
C_2 = t2, p1m2, t3, p11, t4, p2m2, t5, p12, t8, p3m2, t9, p10;
C_3 = t2, p8, t5, p2m1, t6, p9, t7, p3m1, t8, p3m2, t9, p10;
C_4 = t2, p1m2, t3, p11, t4, p2m2, t5, p2m1, t6, p9, t7, p3m1, t8, p3m2, t9, p10;
C_5 = t1, p1m1, t2, p8, t5, p12, t8, p7;
C_6 = t1, p1m1, t2, p1m2, t3, p11, t4, p2m2, t5, p12, t8, p7;
C_7 = t1, p1m1, t2, p8, t5, p2m1, t6, p9, t7, p3m1, t8, p7;
C_8 = t1, p1m1, t2, p1m2, t3, p11, t4, p2m2, t5, p2m1, t6, p9, t7, p3m1, t8, p7.

Marcajul iniţial plasează pe fiecare dintre aceste circuite exact un jeton. Circuitului C_8 îi corespunde cea mai mare întârziere totală, egală cu 25 minute, astfel că durata totală a unui ciclu de funcţionare a grafului marcat din fig. AP6.8.2 este, conform (BT6.5.3), de 25 minute.

Similar se tratează situaţia grafului marcat din fig. AP6.8.3, ai cărui invarianţi *P* calculaţi de ***Petri Net Toolbox*** sunt afişaţi în fig. AP6.8.6.(b). Tratarea completă o lăsăm cititorului drept exerciţiu.


P invariants
Minimal-support P-invariants
m-rank(A)=4 => at most 4 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places (p1m1, p1m2, p2m2, p2m1, p3m1, p3m2, p7, p8, p9, p10, p11, p12)
0 0 0 0 1 1 1 1 ||
0 1 0 1 0 1 0 1 ||
0 1 0 1 0 1 0 1 ||
0 0 1 1 0 0 1 1 ||
0 0 1 1 0 0 1 1 ||
1 1 1 1 0 0 0 0 ||
0 0 0 0 1 1 1 1 ||
1 0 1 0 1 0 1 0 ||
0 0 1 1 0 0 1 1 ||
1 1 1 1 0 0 0 0 ||
0 1 0 1 0 1 0 1 ||
1 1 0 0 1 1 0 0 ||
P invariants
Minimal-support P-invariants
m-rank(A)=4 => at most 4 P-invariants are linearly independent
Linear combinations constructed with these vectors are displayed after ||
Places (p1m1, p1m2, p2m2, p2m1, p3m1, p3m2, p7, p8, p9, p10, p11, p12)
0 0 0 0 1 1 1 1 ||
1 0 1 0 1 0 1 0 ||
0 0 1 1 0 0 1 1 ||
0 0 0 0 1 1 1 1 ||
0 1 0 1 0 1 0 1 ||
0 0 1 1 0 0 1 1 ||
0 0 0 0 1 1 1 1 ||
0 1 0 1 0 1 0 1 ||
1 1 0 0 1 1 0 0 ||
0 0 1 1 0 0 1 1 ||
1 1 1 1 0 0 0 0 ||
1 0 1 0 1 0 1 0 ||

(a) (b)

Fig. AP6.8.6. *Invarianţii de tip P ai reţelelor Petri din* (a) *fig. AP6.8.2 şi* (b) *fig. AP6.8.3.*

Capitolul 7

Modele de tip rețea Petri cu temporizare stohastică

Breviar teoretic

BT7.1. Principiile temporizării stohastice

Dacă duratele de timp asignate tranzițiilor (pozițiilor) unei rețele Petri cu temporizare *T* (sau *P*) au valori aleatoare, se spune că rețeaua respectivă este cu *temporizare stohastică*. În situația în care rețeaua Petri modelează un sistem fizic, este natural a considera că durata de timp asignată unei tranziții (poziții) ce reprezintă o operație, respectă o anumită distribuție de probabilitate. În acest mod, fiecărei tranziții (poziții) temporizate îi este asociată distribuția de probabilitate corespunzătoare duratei de timp asignate. O rețea cu temporizare deterministă poate fi privită drept un caz particular de rețea cu temporizare stohastică (corespunzătoare distribuției constante).

Subliniem faptul că duratele de timp asociate tranzițiilor nu joacă nici un rol în rezolvarea situațiilor conflictuale, spre deosebire de cazul *rețelelor Petri stohastice* care urmează a fi prezentate în capitolul 8. În cazul a două sau mai multe tranziții aflate în conflict într-o rețea Petri cu *temporizare T stohastică*, selectarea tranziției t_i care urmează să se execute se realizează pe baza mecanismului de priorități sau probabilități asignate respectivelor tranziții (similar principiului prezentat în contextul rețelelor cu temporizare *T* deterministă). Apoi se generează durata de timp d_i corespunzătoare legii de repartiție asociată tranziției t_i selectate. Analog situației detaliate în paragraful BT6.1, jetoanele aflate în pozițiile de intrare ale tranziției t_i şi care urmează a fi deplasate la executarea acesteia, rămân rezervate în aceste poziții pe durata d_i. Această durată reprezintă întârzierea manifestată din momentul când tranziția este validată până la executarea acesteia.

Pentru rețelele Petri cu *temporizare P stohastică*, similar rețelelor cu temporizare *P* deterministă, după executarea instantanee a unei tranziții t_i ce precede o poziție temporizată p_j, jetoanele care sunt depuse în p_j prin executarea lui t_i rămân rezervate în p_j pe durata d_j, generată în conformitate cu legea de repartiție asociată poziției p_j. În cazul a două sau mai multe tranziții aflate în conflict, selectarea tranziției care se va executa se realizează pe baza unui mecanism de priorități sau probabilități asignate respectivelor tranziții.

În cazul unui model pentru o structură de conducere a unui sistem cu evenimente discrete, rezultat prin procedeul de sinteză hibridă prezentat în capitolul 4, fiecărei poziții

corespunzătoare unei operaţii i se asociază legea de distribuţie a duratei activităţii de servire (procesare), iar fiecărei poziţii corespunzătoare unei resurse i se asociază legea de distribuţie a duratei activităţii de eliberare a resursei.

BT7.2. Variabile aleatoare - scurtă trecere în revistă

BT7.2.1. Variabile aleatoare

Un concept de bază în teoria probabilităţilor îl reprezintă cel de *experiment*. Mulţimea Ω a tuturor realizărilor posibile ale unui experiment se numeşte *spaţiul eşantioanelor* (eng. *sample space*) acelui experiment. O colecţie de submulţimi ale lui Ω formează *spaţiul evenimentelor* (eng. *event space*) $\mathbb{E}$. *Funcţia de probabilitate P* (eng. *probability function*) asociază fiecărui eveniment probabilitatea de apariţie în urma realizării unui experiment. Tripletul $(\Omega, \mathbb{E}, P)$ se numeşte *câmp de probabilitate* (eng. *probability field*).

Probabilitatea de apariţie a evenimentului *A* în ipoteza că s-a produs evenimentul *B*, notată $P[A \mid B]$, se numeşte *probabilitate condiţională* şi este dată de formula $P[A \mid B] = P[A \cap B] / P[B]$, unde $P[A \cap B]$ notează probabilitatea de apariţie a evenimentului „*A* şi *B*”. Două evenimente *A* şi *B* sunt *independente* dacă şi numai dacă $P[A \cap B] = P[A] \cdot P[B]$ (ceea ce înseamnă că $P[A \mid B] = P[A]$, adică probabilitatea de apariţie a evenimentului *A* nu este influenţată de apariţia evenimentului *B*).

O *variabilă aleatoare* (eng. *random variable*) (*reală*) *X* asociază un număr real $X(\omega)$ fiecărei realizări posibile ω a unui experiment. Astfel, o variabilă aleatoare *X* reprezintă o funcţie definită pe spaţiul eşantioanelor Ω cu valori în mulţimea numerelor reale $\mathbb{R}$, $X : \Omega \to \mathbb{R}$. Pentru consistenţa definiţiei se impune ca mulţimea $\{\omega \in \Omega \mid X(\omega) \le x\}$, notată şi $[X \le x]$, să reprezinte un eveniment (să aparţină spaţiului evenimentelor $\mathbb{E}$) pentru orice $x \in \mathbb{R}$. Mulţimea valorilor pe care le poate lua variabila aleatoare *X*, $\{x \in \mathbb{R} \mid X(\omega) = x,\ \omega \in \Omega\}$, poate fi finită sau numărabilă, caz în care se spune că variabila aleatoare este *discretă*. O variabilă aleatoare pentru care mulţimea valorilor posibile este un interval al dreptei reale (nu neapărat mărginit) este numită *continuă*.

BT7.2.2. Funcţie de repartiţie. Densitate de repartiţie

Se numeşte *funcţia de repartiţie* (sau *de distribuţie*) (eng. *cumulative distribution function - cdf*) a variabilei aleatoare *X* aplicaţia definită prin relaţia:

$$F : \mathbb{R} \to [0, 1], \quad F(x) = P[X \le x]. \tag{BT7.2.1}$$

Funcţia de repartiţie a unei variabile aleatoare *X* este continuă la dreapta în orice punct $x \in \mathbb{R}$, şi are următoarele proprietăţi:

$$\begin{gathered} F(-\infty) = \lim_{x \to -\infty} F(x) = 0, \qquad F(+\infty) = \lim_{x \to +\infty} F(x) = 1, \\ P[x_1 < X \le x_2] = F(x_2) - F(x_1), \quad P[X > x] = 1 - F(x), \\ F(x_1) \le F(x_2), \quad x_1 \le x_2. \end{gathered} \tag{BT7.2.2}$$

Pentru o variabilă aleatoare discretă, $X \in \{x_1, x_2, ...\}$, care ia valoarea x_i cu probabilitatea p_i, $i = 1, 2, \ldots$, se defineşte *densitatea de repartiție*, (eng: *probability density function – pdf*) prin relația:

$$p : \mathbb{R} \to [0, 1],\ p(x) = P[X = x] = \begin{cases} p_i, \text{ dacă } x = x_i, \\ 0, \text{ dacă } x \neq x_i. \end{cases} \quad \text{(BT7.2.3)}$$

Relația de legătură dintre funcția de repartiție şi densitatea de repartiție ale unei variabile aleatoare discrete este:

$$F(x) = \sum_{y \le x} p(y). \quad \text{(BT7.2.4)}$$

Densitatea de repartiție a unei variabile aleatoare continue X se defineşte prin relația:

$$f(x) = \frac{dF(x)}{dx} = \lim_{\Delta x \to 0} \frac{P[x < X \le x + \Delta x]}{\Delta x}, \quad \text{(BT7.2.5)}$$

în ipoteza că derivata aceasta există peste tot în $\mathbb{R}$, cu excepția a cel mult un număr finit de puncte. Relația de legătură dintre funcția de repartiție şi densitatea de repartiție ale unei variabile aleatoare continue este:

$$F(x) = \int_{-\infty}^{x} f(\tau) d\tau. \quad \text{(BT7.2.6)}$$

În practică suntem adeseori puşi în situația de a lua în considerare simultan două sau mai multe variabile aleatoare. Dacă X şi Y sunt două variabile aleatoare reale definite peste acelaşi câmp de probabilitate, atunci perechea $Z = (X, Y)$ este un *vector aleator bidimensional*, $Z : \Omega \to \mathbb{R}^2$.

Se numeşte *funcția de repartiție* a vectorului aleator bidimensional $Z = (X, Y)$ aplicația $F : \mathbb{R}^2 \to [0, 1]$, dată de

$$F_{X,Y}(x, y) = P[X \le x, Y \le y],\ \forall x, y \in \mathbb{R}. \quad \text{(BT7.2.7)}$$

Densitatea de repartiție (eng: *joint density distribution function*) a vectorului aleator bidimensional $Z = (X, Y)$ derivata a doua parțială mixtă (dacă există) a funcției sale de repartiție:

$$f_{X,Y}(x, y) = \frac{\partial^2 F_{X,Y}(x, y)}{\partial x \partial y}. \quad \text{(BT7.2.8)}$$

Funcția de densitate de repartiție bidimensională $f_{X,Y}(x, y)$ are proprietățile:

$$f_{X,Y}(x, y) \ge 0, \quad x, y \in \mathbb{R}; \qquad \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f_{X,Y}(x, y)\, dx\, dy = 1. \quad \text{(BT7.2.9)}$$

Două variabile aleatoare X şi Y sunt *independente* dacă şi numai dacă evenimentele $[X \le x]$ şi $[Y \le y]$ sunt independente pentru orice $x, y \in \mathbb{R}$, deci dacă şi numai dacă

$$F_{X,Y}(x, y) = F_X(x) \cdot F_Y(y),\ \forall x, y \in \mathbb{R}. \quad \text{(BT7.2.10)}$$

BT7.2.3. Funcţii de variabile aleatoare

Presupunem că între două variabile aleatoare X şi Y există relaţia $Y = g(X)$ şi că funcţia de repartiţie şi densitatea de repartiţie a variabilei X sunt cunoscute, fiind notate $F_X(x)$, respectiv $f_X(x)$. Funcţia de repartiţie a variabilei Y, $F_Y(y)$, se poate calcula pe baza relaţiei:

$$F_Y(y) = P[Y \le y] = P[g(X) \le y]. \qquad \text{(BT7.2.11)}$$

Natura funcţiei $g(\cdot)$ determină modul de calcul efectiv al funcţiei $F_Y(y)$. Dacă funcţia g este derivabilă, densitatea de repartiţie a variabilei Y, $f_Y(y)$, poate fi calculată cu ajutorul formulei:

$$f_Y(y) = \sum_i \frac{f_X(x_i)}{|\, g'(x_i)\,|}, \qquad \text{(BT7.2.12)}$$

unde $x_1, x_2, \ldots$ reprezintă rădăcinile ecuaţiei $y = g(x)$ şi $g'(x_i)$ notează valoarea derivatei $dg(x)/dx$ în punctul $x = x_i$.

În general, funcţia de repartiţie $F_Y(y)$ poate fi determinată prin evaluarea integralei:

$$F_Y(y) = \int_{S(y)} f_X(x)\, dx, \qquad \text{(BT7.2.13)}$$

unde $S(y)$ este mulţimea punctelor $\{x \mid g(x) \le y\}$. Complexitatea calculului acestei integrale depinde de complexitatea domeniului $S(y)$, care, la rândul său, depinde de complexitatea funcţiei $g(\cdot)$.

Relaţiile de mai sus se pot extinde în mod natural la cazul funcţiilor care depind de mai multe variabile aleatoare. În particular, dacă $Z = g(X,Y)$ este o funcţie de două variabile aleatoare, atunci

$$F_Z(z) = \int_{S(z)} f_{X,Y}(x,y)\, dx\, dy, \qquad \text{(BT7.2.14)}$$

unde $f_{X,Y}(x,y)$ este *densitatea de repartiţie* a vectorului aleator bidimensional (X,Y) şi $S(z) = \{(x,y) \mid g(x,y) \le z\}$.

De exemplu, dacă X şi Y sunt variabile aleatoare independente ($f_{X,Y}(x,y) = f_X(x) f_Y(y)$) şi $Z = X + Y$, densitatea de repartiţie $f_Z(z) = dF_Z(z)/dz$ este dată de:

$$f_Z(z) = \int_{-\infty}^{\infty} f_X(\tau)\, f_Y(z-\tau)\, d\tau. \qquad \text{(BT7.2.15)}$$

Acest rezultat se poate extinde la suma a *n* variabile aleatoare independente.

BT7.2.4. Caracteristici ale distribuţiilor de probabilitate

BT7.2.4.1. Valoare medie

Pentru variabila aleatoare discretă X care ia valorile $x_1, x_2, \ldots$, cu probabilităţile $p_1, p_2, \ldots$, se defineşte *valoarea medie* (eng: *expectation*) a variabilei aleatoare prin relaţia:

$$\mathrm{M}[X] = \sum_i x_i p_i \overset{not}{=} m_X, \tag{BT7.2.16}$$

dacă seria de mai sus este absolut convergentă, adică $\sum_i |x_i| p_i < +\infty$.

Analog, *valoarea medie* a unei variabile aleatoare continuă X având densitatea de repartiție f, este definită prin relația:

$$\mathrm{M}[X] = \int_{-\infty}^{+\infty} x f(x)\, dx \overset{not}{=} m_X, \tag{BT7.2.17}$$

dacă integrala din membrul doi este convergentă.

Dacă X şi Y sunt variabile aleatoare având valori medii finite, atunci

$$\begin{aligned} &\mathrm{M}[X+Y] = \mathrm{M}[X] + \mathrm{M}[Y], \\ &\mathrm{M}[X \cdot Y] = \mathrm{M}[X] \cdot \mathrm{M}[Y], \quad \text{dacă } X \text{ şi } Y \text{ sunt independente}, \\ &\mathrm{M}[cX] = c\,\mathrm{M}[X], \quad \forall c \in \mathbb{R}. \end{aligned} \tag{BT7.2.18}$$

BT7.2.4.2. Dispersie (varianță)

Dispersia (*varianța*) (eng: *variance*) variabilei aleatoare X care are valoarea medie m_X este dată de:

$$\mathrm{Var}[X] = \mathrm{M}\left[(X - m_X)^2\right] \overset{not}{=} \sigma_X^2, \tag{BT7.2.19}$$

dacă există valoarea medie a variabilei aleatoare $(X - m_X)^2$. Numărul $\sigma_X = \sqrt{\mathrm{Var}[X]}$ se numeşte *deviație standard* (eng: *standard deviation*).

Dispersia unei variabile aleatoare discrete X care ia valorile $x_1, x_2, \ldots$, cu probabilitățile $p_1, p_2, \ldots$, este:

$$\mathrm{Var}[X] = \sum_i \left(x_i - m_X\right)^2 p_i. \tag{BT7.2.20}$$

Dispersia variabilei aleatoare continue X având densitatea de repartiție f este:

$$\mathrm{Var}[X] = \int_{-\infty}^{+\infty} \left(x - m_X\right)^2 f(x) dx. \tag{BT7.2.21}$$

Se pot demonstra următoarele proprietăți:

$$\begin{aligned} &\mathrm{Var}[X] = \mathrm{M}[X^2] - \left(\mathrm{M}[X]\right)^2, \\ &\mathrm{Var}[cX] = c^2\,\mathrm{Var}[X], \quad \forall c \in \mathbb{R}, \\ &\mathrm{Var}[c+X] = \mathrm{Var}[X], \quad \forall c \in \mathbb{R}, \\ &\mathrm{Var}[X+Y] = \mathrm{Var}[X] + \mathrm{Var}[Y], \quad \text{dacă } X \text{ şi } Y \text{ sunt independente.} \\ &P\left[|X - m_X| \geq k\sigma_X\right] \leq \frac{1}{k^2}, \quad k \in \mathbb{R}, \quad k > 0 \quad \text{(inegaltatea lui Cebîşev).} \end{aligned} \tag{BT7.2.22}$$

BT7.2.4.3. Moment de ordin *k*

Se numeşte *moment de ordin k* (eng: *k-th moment*) al variabilei aleatoare X media variabilei X^k,

$$\mu_k[X] = \mathrm{M}[X^k], \tag{BT7.2.23}$$

dacă această valoare medie există.

Pentru o variabilă aleatoare discretă X care ia valorile $x_1, x_2, \ldots$, cu probabilităţile $p_1, p_2, \ldots$, se obţine

$$\mu_k[X] = \sum_i x_i^k p_i, \tag{BT7.2.24}$$

iar pentru o variabilă aleatoare continuă X cu densitatea de repartiţie f,

$$\mu_k[X] = \int_{-\infty}^{+\infty} x^k f(x)dx. \tag{BT7.2.25}$$

Se numeşte *moment centrat de ordin k* (eng: *k-th centered moment*) al variabilei aleatoare X care are valoarea medie m_X, media variabilei $\left(X - m_X\right)^k$, (dacă această valoare medie există)

$$\mu_k^\circ[X] = \mathrm{M}\left[\left(X - m_X\right)^k\right]. \tag{BT7.2.26}$$

Pentru o variabilă aleatoare discretă X se obţine:

$$\mu_k^\circ[X] = \sum_i \left(x_i - m_X\right)^k p_i. \tag{BT7.2.27}$$

iar pentru o variabilă aleatoare continuă X:

$$\mu_k^\circ[X] = \int_{-\infty}^{+\infty} \left(x - m_X\right)^k f(x)dx. \tag{BT7.2.28}$$

Se observă că $\mu_1^\circ[X] = 0$ şi $\mu_2^\circ[X] = \mathrm{Var}[X]$.

BT7.2.4.4. Covarianţă

Se numeşte *covarianţă* (eng: *covariance*) a variabilelor aleatoare X şi Y peste acelaşi câmp de probabilitate numărul (dacă acesta există)

$$\mathrm{Cov}[X,Y] = \mathrm{M}\left[(X - m_x)(Y - m_Y)\right]. \tag{BT7.2.29}$$

Sunt satisfăcute următoarele proprietăţi:

$$\begin{aligned}
&\mathrm{Cov}[X,Y] = \mathrm{Cov}[Y,X], \quad \mathrm{Cov}[X,X] = \mathrm{Var}[X],\\
&\mathrm{Cov}[X,Y] = \mathrm{M}[XY] - \mathrm{M}[X]\mathrm{M}[Y],\\
&\mathrm{Cov}[X,Y] = 0, \quad \text{dacă } X \text{ şi } Y \text{ sunt independente},\\
&\mathrm{Cov}[X+Y,Z] = \mathrm{Cov}[X,Z] + \mathrm{Cov}[Y,Z].
\end{aligned} \tag{BT7.2.30}$$

Coeficientul de corelație (eng: *correlation coefficient*) a două variabile aleatoare X şi Y ale căror deviații standard sunt σ_X, respectiv σ_Y, este definit prin:

$$\rho_{X,Y} = \frac{\mathrm{Cov}[X,Y]}{\sigma_X \sigma_Y}. \qquad \text{(BT7.2.31)}$$

O valoare pozitivă a lui $\rho_{X,Y}$ înseamnă că variabilele aleatoare X şi Y prezintă variații în același sens. O valoare negativă a lui $\rho_{X,Y}$ înseamnă că variabilele aleatoare X şi Y prezintă variații în sensuri contrare.

Dacă modulul coeficientului de corelație $|\rho_{X,Y}|$ are o valoare apropiată de 1, înseamnă că între variabilele aleatoare X şi Y există a puternică dependență de factură liniară. Dacă modulul coeficientului de corelație $|\rho_{X,Y}|$ are o valoare apropiată de 0, înseamnă că dependența de factură liniară între variabilele aleatoare X şi Y este redusă.(fără ca aceasta să excludă posibilitatea unei dependențe de factură neliniară). Dacă modulul coeficientului de corelație $|\rho_{X,Y}|$ are exact valoarea 1, înseamnă că între variabilele aleatoare X şi Y există o dependență liniară strictă.

Astfel, dacă modulul coeficientului de corelație $|\rho_{X,Y}|$ are o valoare apropiată de 1, atunci se poate efectua o analiză de regresie, căutând o dependență între valorile variabilelor X şi Y de factură liniară, $Y(X) = AX + B$. Valorile numerice ale parametrilor A şi B se obțin ca soluție a unei probleme de minimizare formulată în sensul celor mai mici pătrate.

BT7.2.5. Legea numerelor mari. Legi limită

Există mai multe *legi ale numerelor mari* care stabilesc legătura dintre frecvența de apariție a unui eveniment şi probabilitate, utilizate pentru a justifica rezultatele obținute prin simulare şi a evalua erorile de estimare. Două dintre aceste legi sunt prezentate în continuare.

Teorema lui Bernoulli. Fie α numărul de apariții ale unui eveniment A în n experimente independente şi p probabilitatea de apariție a lui A în fiecare dintre aceste experiențe. Dacă f_n este frecvența relativă definită prin egalitatea $f_n = \alpha / n$, atunci şirul de variabile aleatoare $\{f_n\}_{n\in\mathbb{N}}$ converge în probabilitate la p, adică

$$\lim_{n\to\infty} P\left[\left|f_n - p\right| \geq \varepsilon\right] = 0. \qquad \text{(BT7.2.32)}$$

Din punct de vedere practic, această teoremă arată că probabilitatea de apariție a evenimentului A poate fi aproximată oricât de bine prin frecvența relativă determinată exeprimental atunci când numărul n de experimente efectuate este suficient de mare.

Legea numerelor mari. Fie variabilele aleatoare independente $X_1, X_2, ..., X_n$, având aceeaşi distribuție de probabilitate, cu valoarea medie $\mu = \mathrm{M}[X_k]$ şi dispersia $\sigma^2 = \mathrm{Var}[X_k]$ (finite). Considerând variabila aleatoare $S_n = X_1 + X_2 + ... + X_n$, pentru orice $\varepsilon > 0$ este satisfăcută relația:

$$\lim_{n\to\infty} P\left[\left|\frac{S_n}{n} - \mu\right| \geq \varepsilon\right] = 0. \qquad \text{(BT7.2.33)}$$

Tab. BT7.2.1. *Distribuții uzuale de probabilitate*

Numele distribuției	**Simbol**	**Densitate de probabilitate**	**Parametri**	**Valoare medie**	**Varianță**	**Domeniul valorilor variabilei aleatoare**
Uniformă discretă	$\boldsymbol{U}_d(N)$	$f(n)=\frac{1}{N},\quad n\in\{1,2,...,N\}$	$N\in\mathbb{N}^*$	$\frac{N+1}{2}$	$\frac{N^2-1}{12}$	$X\in\{1,2,...,N\}$
Geometrică	$\boldsymbol{G}_d(p)$	$f(n)=p(1-p)^{n-1},\quad n\in\mathbb{N}^*$	$p\in(0,1)$	$\frac{1-p}{p}$	$\frac{1-p}{p^2}$	$X\in\mathbb{N}^*$
Poisson	$\boldsymbol{P}(\lambda)$	$f(n)=\frac{\lambda^n}{n!}\mathrm{e}^{-\lambda},\quad n\in\mathbb{N}^*$	$\lambda>0$	λ	λ	$X\in\mathbb{N}^*$
Binomială	$\boldsymbol{Bin}(p,N)$	$f(k)=C_N^k p^k(1-p)^{N-k},\quad k\in\{0,1,2,...,N\}$	$p\in(0,1)$, $N\in\mathbb{N}^*$	$\frac{N}{p}$	$\frac{Np}{1-p}$	$X\in\{0,1,2,...,N\}$
Uniformă continuă	$\boldsymbol{U}(a,b)$	$f(x)=\frac{1}{b-a},\quad x\in[a,b]$	$a,b\in\mathbb{R}$, $a<b$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$	$X\in[a,b]$
Exponențială	$\boldsymbol{E}(\lambda)$	$f(x)=\lambda\mathrm{e}^{-\lambda x},\quad x\geq 0$	$\lambda>0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$	$X\in[0,+\infty)$
Normală	$\boldsymbol{N}(\mu,\sigma)$	$f(x)=\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}},\quad x\in\mathbb{R}$	$\mu\in\mathbb{R}$, $\sigma>0$	μ	σ^2	$X\in\mathbb{R}$
Gamma	$\boldsymbol{G}(\alpha,\beta)$	$f(x)=\frac{1}{\beta^\alpha\Gamma(\alpha)}x^{\alpha-1}\mathrm{e}^{-\frac{x}{\beta}},\quad x\geq 0$	$\alpha,\beta>0$	$\alpha\beta$	$\alpha\beta^2$	$X\in[0,+\infty)$
Erlang	$\boldsymbol{Er}(k,\beta)$	$f(x)=\frac{1}{\beta^k(k-1)!}x^{k-1}\mathrm{e}^{-\frac{x}{\beta}},\quad x\geq 0$	$k\in\mathbb{N}^*$, $\beta>0$	$k\beta$	$k\beta^2$	$X\in[0,+\infty)$
χ^2	$\boldsymbol{\chi}^2(\nu)$	$f(x)=\frac{1}{2^{\nu/2}\Gamma(\nu/2)}x^{\frac{\nu-2}{2}}\mathrm{e}^{-\frac{x}{2}},\quad x\geq 0$	$\nu\in\mathbb{N}^*$	ν	2ν	$X\in[0,+\infty)$
Beta	$\boldsymbol{B}(\alpha,\beta)$	$f(x)=\frac{1}{\mathrm{B}(\beta,\alpha)}x^{\beta-1}(1-x)^{\alpha-1},\quad x\in[0,1]$	$\alpha,\beta>0$	$\frac{\beta}{\alpha+\beta}$	$\frac{\alpha\beta(\alpha+\beta)^{-2}}{(\alpha+\beta+1)}$	$X\in[0,1]$

Din punct de vedere practic, această lege arată că valoarea medie μ a unei variabile aleatoare X poate fi aproximată oricât de bine prin media aritmetică a unui număr n suficient de mare de realizări (experimentale) ale acesteia, $X_1, X_2, ..., X_n$.

BT7.2.6. Distribuții frecvent folosite în temporizarea stohastică

Tabelul BT7.2.1 prezintă sumar câteva dintre distribuțiile de probabilitate frecvent întâlnite în probleme de modelare a sistemelor dinamice cu evenimente discrete, împreună cu caracteristicile acestora.

BT7.3. Procese stohastice

Un *proces stohastic X* (eng: *stochastic process*) reprezintă o familie de variabile aleatoare definite pe un câmp de probabilitate $(\Omega, \mathbb{E}, P)$ indexate după un parametru $t \in \mathbb{T}$ (care, de regulă, are semnificația de timp), $X : \mathbb{T} \times \Omega \to \mathbb{R}$. Pentru $t \in \mathbb{T}$ fixat, $X_t(\omega)$ este o variabilă aleatoare, $X_t : \Omega \to \mathbb{R}$, numită *eşantion* (eng. *sample*) al procesului stohastic. Pentru $\omega \in \Omega$ fixat se obține o funcție definită pe $\mathbb{T}$, numită *realizare* sau *traiectorie* (eng. *sample path*) a procesului stohastic. Mulțimea tuturor valorilor posibile ale unui proces stohastic, $\mathcal{S} = \{X_t(\omega) | t \in \mathbb{T}, \omega \in \Omega\}$, se numeşte *spațiul stărilor* (eng. *state space*) procesului respectiv.

Procesele stohastice pot fi clasificate după natura discretă sau continuă a spațiului stărilor şi a spațiului parametrilor. Procesele stohastice pentru care spațiul stărilor este discret (mulțime finită sau numărabilă) se numesc *lanțuri* (eng. *chain*). Dacă mulțimea $\mathbb{T}$ pe care este definită variabila temporală t este discretă se spune că procesul stohastic $\{X(t)\}$ este *discret în timp* (eng. *discrete-time stochastic process*); în acest caz se consideră $\mathbb{T}=\{0,1,2,...\} \subseteq \mathbb{N}$ iar procesul stohastic discret este notat $\{X_k\}$, $k = 0,1,2,...$. Un proces stohastic pentru care $\mathbb{T} = \mathbb{R}_+$ este referit drept proces stohastic *continuu în timp* (eng. *continuous-time stochastic process*).

Pentru orice $t \in \mathbb{T}$, *funcția de repartiție* a procesului stohastic X se defineşte ca fiind funcția de repartiție a variabilei aleatoare X_t:

$$P_X(x,t) = P[X_t < x], \quad x \in \mathbb{R}, \quad t \in \mathbb{T}. \tag{BT7.3.1}$$

Similar se defineşte *densitatea de repartiție* a procesului stohastic X ca reprezentând densitatea de repartiție a variabilei aleatoare X_t, pentru orice $t \in \mathbb{T}$.

Valoarea medie a unui proces stohastic X este funcția $m_X(t)$ care, pentru fiecare moment $t \in \mathbb{T}$, reprezintă valoarea medie a variabilei aleatoare X_t:

$$m_X : \mathbb{T} \to \mathbb{R}, \quad m_X(t) = M[X_t], \quad t \in \mathbb{T}. \tag{BT7.3.2}$$

Funcția de autocorelație a procesului stohastic X este funcția de două argumente $R_X(t,t')$ egală cu covarianța variabilelor aleatoare X_t şi $X_{t'}$

$$R_X : \mathbb{T} \times \mathbb{T} \to \mathbb{R}, \quad R_X(t,t') = \mathrm{Cov}[X_t, X_{t'}], \quad t,t' \in \mathbb{T}. \tag{BT7.3.3}$$

Un proces stohastic se numeşte *staționar* (eng. *stationary*) dacă funcția de repartiție a procesului stohastic $X_t(\omega)$ coincide cu funcția de repartiție a procesului $X_{t+\tau}(\omega)$, pentru orice $t,\tau \in \mathbb{T}$, adică

$$P_X(x,t) = P_X(x,t+\tau), \quad \forall x \in \mathbb{R}, \quad t,\tau \in \mathbb{T}. \tag{BT7.3.4}$$

Dacă un proces stohastic este staționar, atunci funcția sa de repartiție şi, implicit, densitatea de repartiție, sunt independente de timp. De asemenea, pentru un proces stohastic staționar, valoarea medie este constantă iar funcția de corelație depinde numai de diferența dintre argumentele sale, $R_X(t,t') = R_X(\tau)$, cu $\tau = t' - t$, fiind o funcție pară, $R_X(\tau) = R_X(-\tau)$.

Un proces stohastic staționar se numeşte *ergodic* (eng. *ergodic*) în cazul în care caracteristicile sale m_X şi $R_X(\tau)$ pot fi definite drept valori medii corespunzătoare unei singure realizări de pe o durată de timp suficient de mare. Orice realizare pe o durată de timp suficient de mare a unui proces stohastic staționar şi ergodic poate fi utilizată pentru caracterizarea procesului însuşi.

BT7.4. Modele de tip rețea Petri cu temporizare stohastică

În general, modelarea unui sistem fizic printr-o rețea Petri cu temporizare stohastică se realizează în scopul evaluării performanțelor de regim permanent ale sistemului, presupunând că se poate estima „comportarea medie" a sistemului pe baza observării (simulării) modelului pe un interval de timp suficient de lung. Această presupunere implică ipoteza ergodicității modelului: la limită, atunci când intervalul de observare (simulare) tinde la infinit, estimările valorilor medii ale indicilor de performanță studiați (de interes) tind aproape sigur la valorile medii teoretice.

Astfel, performanțele sistemului fizic se pot estima pe baza analizei a două procese stohastice asociate modelului de tip rețea Petri cu temporizare stohastică, şi anume procesul de executare a tranzițiilor şi cel de marcaj. De regulă, se presupune că aceste două procese stohastice sunt ergodice, astfel încât pe baza modelelor de tip rețea Petri cu temporizare stohastică se pot analiza performanțele sistemelor fizice modelate (gradele de utilizare a resurselor, număr mediu de clienți serviți în unitatea de timp, duratele medii ale ciclurilor de producție, etc.). Fiecare din aplicațiile **AP7.1** – **AP7.5** ilustrează aceste aspecte.

Aplicații

AP7.1.

Fie un server care serveşte două tipuri de clienți în paralel, modelat prin rețeaua Petri cu temporizare *T* stohastică din fig. AP7.1.1. Servirea primului tip de clienți este modelată de subrețeaua din partea stângă a figurii (cu nodurile p_1, t_1, p_2, t_2), iar servirea celui de-al doilea tip de clienți de subrețeaua din partea dreaptă a figurii (cu nodurile p_1, t_3, p_3, t_4). Durata de

timp asignată tranziției t_i, $i = \overline{1,4}$, are distribuție exponențială cu rata λ_i [unități de timp $^{-1}$]. Se consideră următoarele valori numerice: $\lambda_1 = \lambda_4 = 1$, $\lambda_2 = \lambda_3 = 10$.

Pentru fiecare din situațiile de mai jos să se determine, prin simulare în mediul ***Petri Net Toolbox***, numărul de clienți de fiecare tip serviți complet într-un interval de timp de 10.000 [unități de timp]:

(i) probabilitea de servire a primului tip de clienți este egală cu probabilitea de servire a celui de-al doilea tip de clienți;

(ii) probabilitea de servire a primului tip de clienți este 30%, iar probabilitea de servire a celui de-al doilea tip de clienți este 70%.

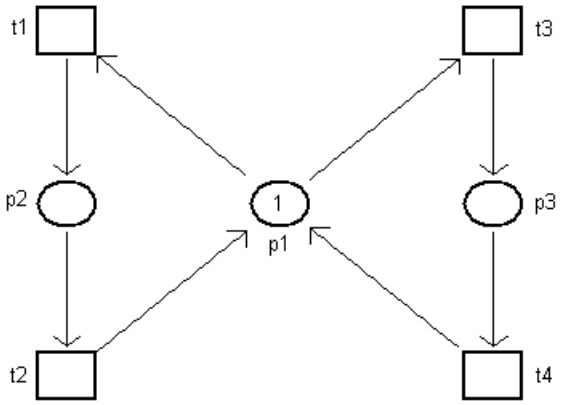


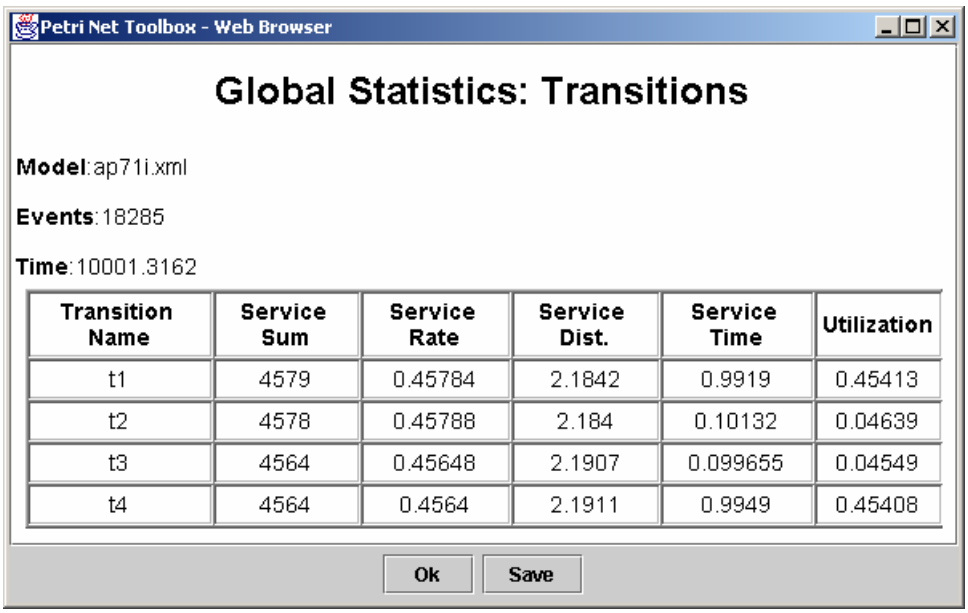
Fig. AP7.1.1. *Reprezentarea grafică a rețelei Petri cu temporizare T studiată în AP7.1.*

Soluție

Pentru a modela cele două situații indicate în enunț, probabilitățile de servire corespunzătoare celor două tipuri de clienți se asociază tranzițiilor t_1 şi t_3 (aflate în conflict).

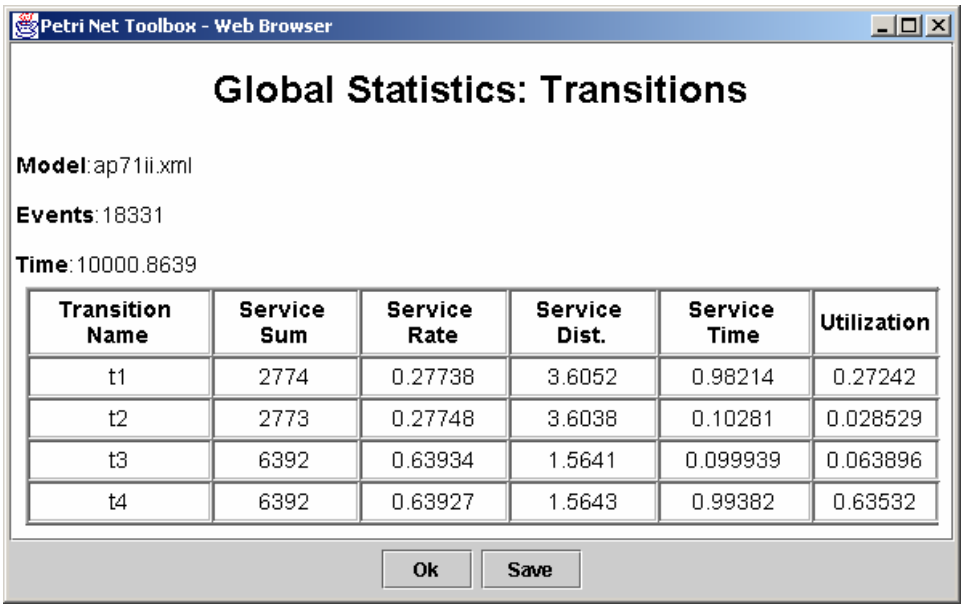
(i). Rezultatele obținute în urma simulării funcționării sistemului timp de 10.000 [unități de timp] sunt prezentate în fig. AP7.1.2 pentru cazul în care probabilitatea de servire a primului tip de clienți este egală cu probabilitatea de servire a celui de-al doilea tip de clienți. Numărul de clienți de primul tip serviți complet este dat de indicatorul ***Service Sum*** pentru tranziția t_2 (având valoarea 4.578), iar numărul de clienți de al doilea tip serviți complet este dat de indicatorul ***Service Sum*** pentru tranziția t_4 (având valoarea 4.564). Se observă că, în acest caz, au fost serviți aproximativ acelaşi număr de clienți din fiecare tip.

(ii). Pentru cazul în care probabilitatea de servire a primului tip de clienți este 30% şi probabilitatea de servire a celui de-al doilea tip de clienți este 70%, rezultatele obținute în urma simulării funcționării sistemului timp de 10.000 [unități de timp] sunt prezentate în fig. AP7.1.3. Numărul de clienți din primul tip serviți complet este 2.773, iar numărul de clienți din al doilea tip serviți complet este 6.392. Se observă că, aşa cum era de aşteptat, raportul dintre cele două numere este aproximativ egal cu raportul dintre probabilitățile de servire.



Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	4579	0.45784	2.1842	0.9919	0.45413
t2	4578	0.45788	2.184	0.10132	0.04639
t3	4564	0.45648	2.1907	0.099655	0.04549
t4	4564	0.4564	2.1911	0.9949	0.45408

Fig. AP7.1.2. *Indicatorii globali pentru tranzițiile rețelei Petri din fig. AP7.1.1 cu probabilități egale de executare a tranzițiilor t_1 şi t_3.*



Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	2774	0.27738	3.6052	0.98214	0.27242
t2	2773	0.27748	3.6038	0.10281	0.028529
t3	6392	0.63934	1.5641	0.099939	0.063896
t4	6392	0.63927	1.5643	0.99382	0.63532

Fig. AP7.1.3. *Indicatorii globali pentru tranzițiile rețelei Petri din fig. AP7.1.1 cu probabilități diferite de executare a tranzițiilor t_1 (30%) şi t_3 (70%).*

Observație: Rețeaua Petri din fig. AP7.1.1 cu temporizarea stohastică *T* considerată, modelează funcționarea unui server partajat paralel în cazul când fiecare client necesită două operații de servire consecutive (servirea fiecărui client se realizează în două faze). La aceleaşi rezultate s-ar fi ajuns dacă am fi utilizat modelul cu temporizare *P* din fig. AP7.1.4, în care duratele de timp asignate pozițiilor p_2, p_3, p_4 şi p_5 au distribuție exponențială cu ratele $\mu_2 = \mu_5 = 1$, respectiv $\mu_3 = \mu_4 = 10$ [unități de timp $^{-1}$] (adică aceleaşi valori ce au fost considerate pentru temporizarea stohastică de tip *T* a rețelei din fig. AP7.1.1).

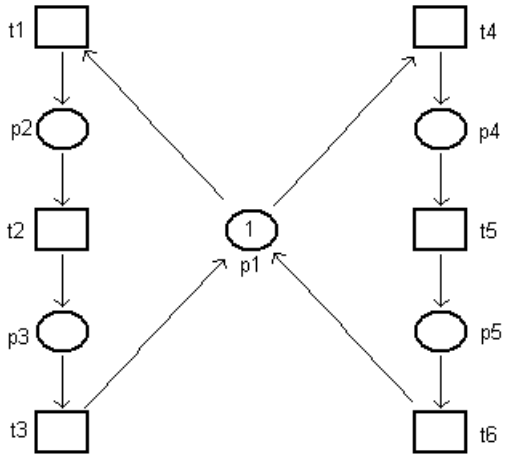


Fig. AP7.1.4. *Rețeaua Petri cu temporizare P asociată celei din fig. AP7.1.1.*

Aşa cum era de aşteptat, valorile indicatorilor globali referitori la pozițiile temporizate stochastic (fig. AP7.1.5) furnizează aceleaşi informații ca şi indicatorii globali referitori la tranzițiile temporizate stochastic (fig. AP7.1.2). De exemplu, numărul de clienți de primul tip serviți complet este dat acum de indicatorul ***Throughput Sum*** pentru poziția p_3 (având valoarea 4.525), iar numărul de clienți de al doilea tip serviți complet este dat de indicatorul ***Throughput Sum*** pentru poziția p_5 (având valoarea 4.561).

Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap7_1_p.xml

Events:27260

Time:10000.0266

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
p1	9086	0.9086	1.1006	9087	0.9088	1.1004	0	0
p2	4526	0.4526	2.2095	4526	0.4526	2.2095	0.99685	0.45117
p3	4526	0.4526	2.2095	4525	0.45271	2.2089	0.099727	0.045126
p4	4561	0.4561	2.1925	4561	0.45615	2.1923	0.098682	0.045009
p5	4561	0.4561	2.1925	4561	0.45615	2.1923	1.0057	0.45869

Ok Save

Fig. AP7.1.5. *Indicatorii globali pentru pozițiile rețelei Petri din fig. AP7.1.4 cu probabilități egale de executare a tranzițiilor t_1 şi t_3.*

AP7.2.

Se consideră sistemul de fabricație din fig. AP7.2.1, alcătuit dintr-un robot R care serveşte, partajat, maşinile identice M_1 şi M_2, în scopul încărcării acestora. Ambele maşini se descarcă automat. Robotul rămâne alocat până la finalizarea prelucrării unei piese pe oricare din maşini pentru a menține fixată piesa respectivă pe maşina care efectuează prelucrarea. Pe M_1 se prelucrează piese de tipul 1 dintr-un lot L_1, pentru care durata de încărcare cu ajutorul lui R este uniform distribuită între 2 şi 4 minute, iar durata de prelucrare (incluzând eliberarea lui M_1 şi R) este uniform distribuită între 8 şi 12 minute. În mod similar, pe M_2 se prelucrează piese de tipul 2 dintr-un lot L_2, pentru care durata de încărcare cu R este uniform distribuită între 3 şi 7 minute, iar durata de prelucrare (incluzând eliberarea lui M_2 şi R) este uniform distribuită între 17 şi 25 minute.

Se consideră că există un număr suficient de mare de piese din ambele tipuri şi că robotul va proceda la încărcarea unei noi piese neprelucrate de îndată ce este liber. Pentru robot nu există nici un mecanism de alegere preferențială a tipului de piesă pe care îl încarcă (din lotul L_1 sau L_2), cele două tipuri de piese având probabilitățile egale de servire.

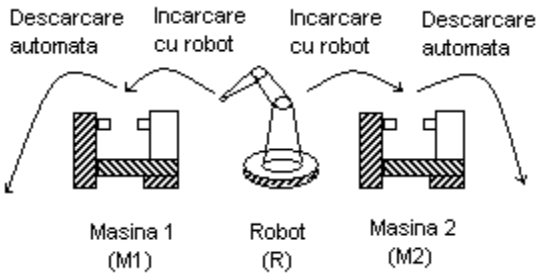


Fig. AP7.2.1. *Sistemul flexibil de fabricație utilizat în AP7.2.*

1. Să se construiască rețeaua Petri cu temporizare stohastică de tip *T* care modelează funcționarea fluxului tehnologic, în conformitate cu descrierea de mai sus.
2. Să se utilizeze modelul construit la punctul 1 pentru analiza în mediul ***Petri Net Toolbox*** a funcționării fluxului tehnologic pe un interval de timp de 2.000 ore, precizând următoarele elemente caracteristice:
    1. numărul de piese prelucrate pe M_1 şi respectiv pe M_2;
    2. timpul mediu necesar procesării unei piese din L_1 respectiv L_2 de la începutul transportului cu R a piesei brute, până la descărcarea automată a piesei finite;
    3. gradul de utilizare a maşinilor M_1 şi M_2;
    4. gradul de utilizare a robotului R.

Soluție

1. Modelul de tip rețea Petri, cu temporizare stohastică de tip *T*, care modelează funcționarea sistemului de fabricație investigat este prezentat în fig. AP7.2.1. Tranzițiile t_2 şi t_3 modelează operațiile de încărcare a unei piese din lotul L_1 pe maşina M_1, respectiv de prelucrare a piesei pe această maşină. Similar, pentru o piesă din lotul L_2, tranzițiile t_5 şi t_6 modelează operațiile de încărcare pe maşina M_2, respectiv de prelucrare a piesei pe această maşină. Drept urmare a

acestor semnificații fizice, duratele de timp asignate acestor tranziții sunt precizate sub forma distribuțiilor uniforme din enunț. Tranzițiile t_1 şi t_4 modelează alegerea tipului de piesă ce va fi încărcată pe una dintre maşini (care nu necesită temporizare, desfăşurându-se instantaneu). Tranzițiilor t_1 şi t_4 li se alocă probabilități egale de executare.

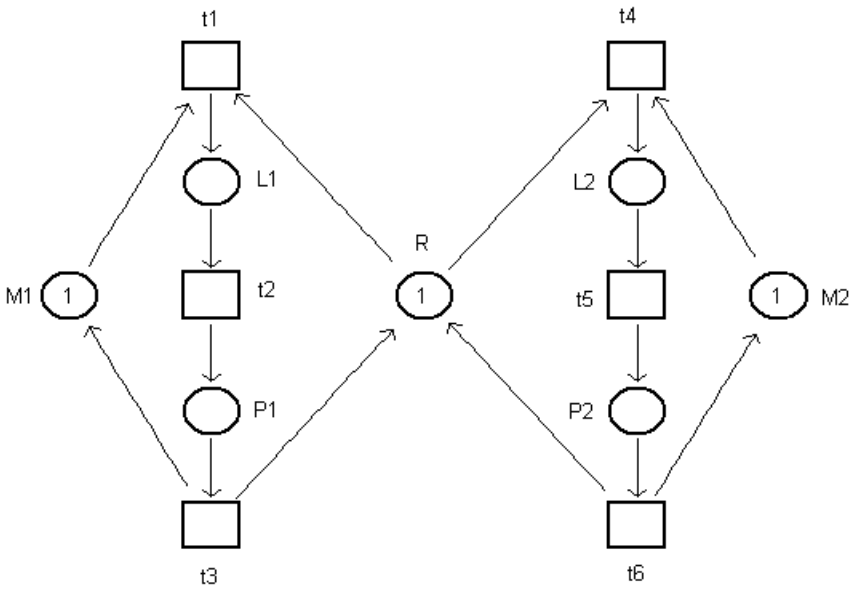


Fig. AP7.2.2. *Modelul de tip rețea Petri al sistemului de fabricație din fig. AP7.2.1.*

2. Prin simularea în mediul ***Petri Net Toolbox*** a funcționării sistemului de fabricație pe o durată de 2.000 ore se obțin indicatorii globali (referitori la tranzițiile şi pozițiile modelului) care sunt prezentați în fig. AP7.2.3.

Pe baza acestor indicatori se obțin următoarele performanțe ale sistemului de fabricație:

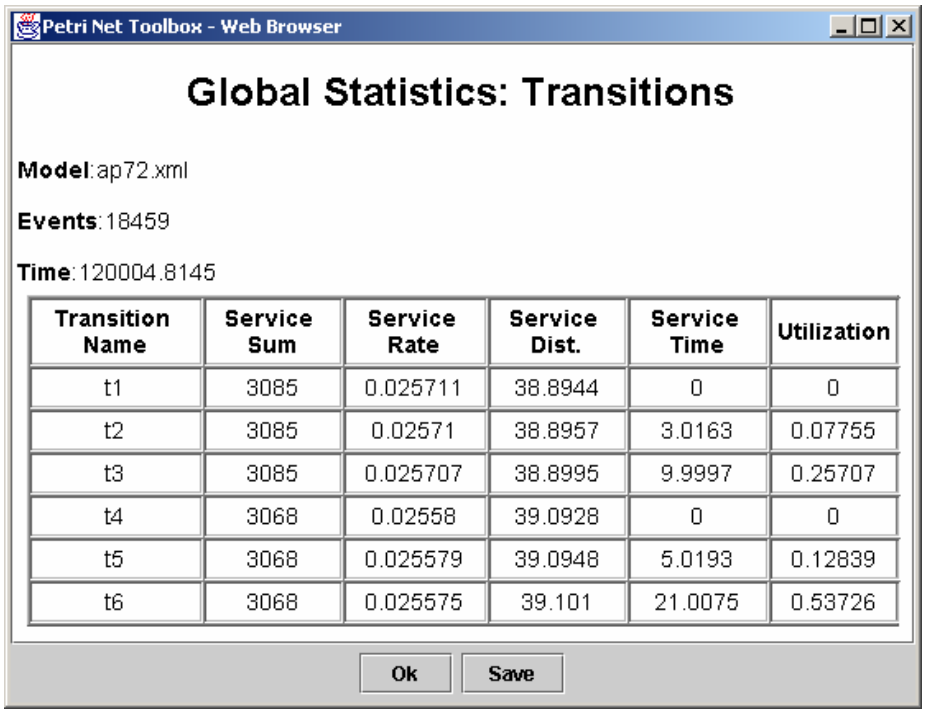
(i) numărul de piese prelucrate complet
- pe maşina M_1: ***Service Sum***(t_3) = 3.085,
- pe maşina M_2: ***Service Sum***(t_6) = 3.0853.068;

(ii) timpul mediu necesar procesării unei piese
- din lotul L_1: ***Service Time***(t_2) + ***Service Time***(t_3) = 13 min.,
- din lotul L_2: ***Service Time***(t_5) + ***Service Time***(t_6) = 26 min.;

(iii) gradul de utilizare
- a maşinii M_1: 1 – ***Queue Length***(M1) = ***Utilization***(t_2) + ***Utilization***(t_3) = 0.335,
- a maşinii M_2: 1 – ***Queue Length***(M2) = ***Utilization***(t_5) + ***Utilization***(t_6) = 0.665;

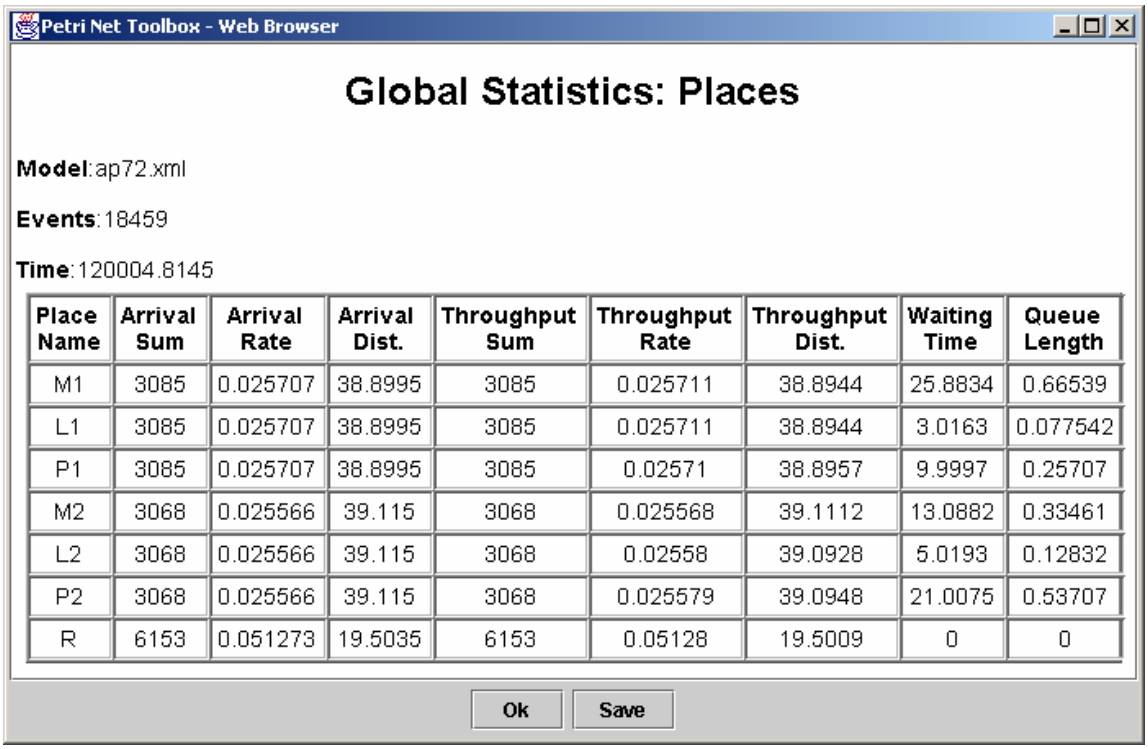
(iv) gradul de utilizare a robotului R: 1 – ***Queue Length***(R) = 1.

Cititorului i se sugerează să compare analiza de mai sus cu situația când rețeaua Petri din fig. AP7.2.2 ar fi cu temporizare *T* deterministă, tranzițiilor t_2, t_3, t_5 şi t_6 fiindu-le asignate (corespunzător) valorile medii ale distribuțiilor uniforme menționate în enunț.



Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	3085	0.025711	38.8944	0	0
t2	3085	0.02571	38.8957	3.0163	0.07755
t3	3085	0.025707	38.8995	9.9997	0.25707
t4	3068	0.02558	39.0928	0	0
t5	3068	0.025579	39.0948	5.0193	0.12839
t6	3068	0.025575	39.101	21.0075	0.53726

(a)



Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
M1	3085	0.025707	38.8995	3085	0.025711	38.8944	25.8834	0.66539
L1	3085	0.025707	38.8995	3085	0.025711	38.8944	3.0163	0.077542
P1	3085	0.025707	38.8995	3085	0.02571	38.8957	9.9997	0.25707
M2	3068	0.025566	39.115	3068	0.025568	39.1112	13.0882	0.33461
L2	3068	0.025566	39.115	3068	0.02558	39.0928	5.0193	0.12832
P2	3068	0.025566	39.115	3068	0.025579	39.0948	21.0075	0.53707
R	6153	0.051273	19.5035	6153	0.05128	19.5009	0	0

(b)

Fig. AP7.2.3. *Indicatorii globali pentru*
(a) *tranziţiile şi* (b) *poziţiile reţelei Petri din fig. AP7.2.2.*

AP7.3.

Ne plasăm în contextul sistemului de fabricație al cărui model a fost construit la **AP4.2** punctul 1.(i). Duratele activităților desfășurate în sistemul de fabricație, exprimate în unități de timp, au distribuții uniforme după cum urmează: d_{M1} în intervalul [35, 45], d_{M2} în intervalul [80, 90] pentru procesarea pe M_1, respectiv M_2 şi d_{R1} în intervalul [15, 25], d_{R2} în intervalul [15, 25] pentru descărcarea cu ajutorul lui R a lui M_1, respectiv M_2. Timpul necesitat de eliberarea resurselor M_1, M_2 şi respectiv R este considerat neglijabil. Timpul necesitat de eliberarea unei palete d_p este uniform distribuit în intervalul [18, 22].

1. Funcționarea sistemului de fabricație este modelată de o rețea Petri cu temporizare *P* stohastică. Să se precizeze cum poate fi obținut acest model pornind de la rețeaua Petri netemporizată reprezentată în fig. AP4.2.4.
2. Să se utilizeze modelul construit la punctul 1 pentru analiza în mediul ***Petri Net Toolbox*** a funcționării sistemului pe un interval de timp corespunzător prelucrării complete a 10.000 de piese, precizând:
   (i) durata medie necesară fabricării unui produs finit;
   (ii) gradul de utilizare a robotului R;
   (iii) gradul de utilizare a maşinilor M_1 şi M_2.

Soluție

Petri Net Toolbox - Web Browser

Global Statistics: Places

Model:ap7_3.xml

Events:60007

Time:961429.7758

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
PM1	10002	0.010403	96.1238	10002	0.010404	96.1198	42.8941	0.44624
TR1	10002	0.010403	96.1238	10002	0.010403	96.1222	19.9702	0.20775
WB	10002	0.010403	96.1238	10001	0.010402	96.1334	89.4887	0.93088
PM2	10001	0.010402	96.1334	10000	0.010401	96.143	96.1365	0.99993
TR2	10000	0.010401	96.143	10000	0.010401	96.143	19.9186	0.20718
P	10000	0.010401	96.143	10002	0.010404	96.116	19.9952	0.20802
M1	10002	0.010403	96.1238	10002	0.010403	96.1238	53.2296	0.55376
D	10001	0.010402	96.1334	10002	0.010403	96.1238	102.7677	1.0691
M2	10000	0.010401	96.143	10001	0.010402	96.1319	0.0064688	6.7289e-005
R	20002	0.020804	48.0667	20002	0.020805	48.0659	28.1223	0.58507

Ok Save

Fig. AP7.3.1. *Indicatorii globali pentru pozițiile rețelei Petri din fig. AP4.2.4 considerată cu temporizare stohastică de tip P.*

1. Modelul de tip rețea Petri netemporizată pentru structura de conducere SC1 este prezentat în fig. AP4.2.4. Pentru obținerea modelului temporizat, distribuțiile corespunzătoare duratelor

de timp ale activităților (d_{M1}, d_{M2}, d_{R1} şi d_{R2}) se asignează pozițiilor care modelează operațiile (PM1, PM2, TR1 şi, respectiv, TR2). Distribuția corespunzătoare duratei de eliberare a unei palete, d_p, se asignează poziției care modelează disponibilitatea paletelor (P).

2. Prin simularea în mediul ***Petri Net Toolbox*** a funcționării sistemului de fabricație pe o durată corespunzătoare prelucrării complete a 10.000 de piese se obțin indicatorii globali (referitori la pozițiile modelului) prezentați în fig. AP7.3.1.

Pe baza informațiilor din fig. AP7.3.1 se pot determina următoarele performanțe:

(i) durata medie necesară fabricării unui produs finit: ***Arrival Distance***(P) = 96,16 [unități de timp];

(ii) gradul de utilizare a robotului R: 1 – ***Queue Length***(R) = ***Queue Length***(TR1) + ***Queue Length***(TR2) = 0,415;

(iii) gradul de utilizare al maşinilor:

- pentru M_1: 1 – ***Queue Length***(M1) = ***Queue Length***(PM1) = 0,999,
- pentru M_2: 1 – ***Queue Length***(M2) = ***Queue Length***(PM2) = 0,446.

AP7.4.

Ne plasăm în contextul sistemului de fabricație al cărui model, notat SC1, a fost construit la **AP4.2** punctul 1.(i). Duratele activităților desfăşurate în sistemul de fabricație, exprimate în unități de timp, sunt $d_{M1} = 40$, $d_{M2} = 85$ pentru procesarea pe M_1, respectiv M_2, şi $d_{R1} = 20$, $d_{R2} = 20$ pentru descărcarea cu ajutorul lui R a lui M_1, respectiv M_2. Timpul necesitat de eliberarea resurselor M_1, M_2 şi respectiv R este considerat neglijabil.

Se consideră că durata necesară eliberării paletelor d_P are distribuție exponențială de medie μ ajustabilă în intervalul [20, 30] unități de timp. Folosind meniul ***Design*** al mediului ***Petri Net Toolbox*** şi considerând un timp de funcționare de 50.000 unități de timp, să se reprezinte grafic, în funcție de valoarea μ, următoarele caracteristici ale sistemului de fabricație:

(i) durata medie necesară fabricării unui produs finit;

(ii) durata medie de ocupare cu piesă a maşinii M_1;

(iii) durata medie de ocupare cu piesă a maşinii M_2.

Soluție

Se consideră modelul de tip rețea Petri temporizată *P* pentru care duratele operațiilor de prelucrare pe maşini şi de transport au valorile constante indicate în enunț.

(i). Utilizând facilitatea ***Design*** oferită de mediul ***Petri Net Toolbox*** şi considerând drept parametru valoarea medie μ a distribuției exponențiale asignate poziției care modelează disponibilitatea paletelor, se obține reprezentarea grafică din fig. AP7.4.1. Se constată că durata medie de prelucrare completă a unei piese se modifică foarte puțin atunci când μ ia valori în intervalul suficient de larg [20, 30]. Practic, variațiile sunt foarte mici, însemnând (89,6 – 89,2)/89,6 = 0,44%.

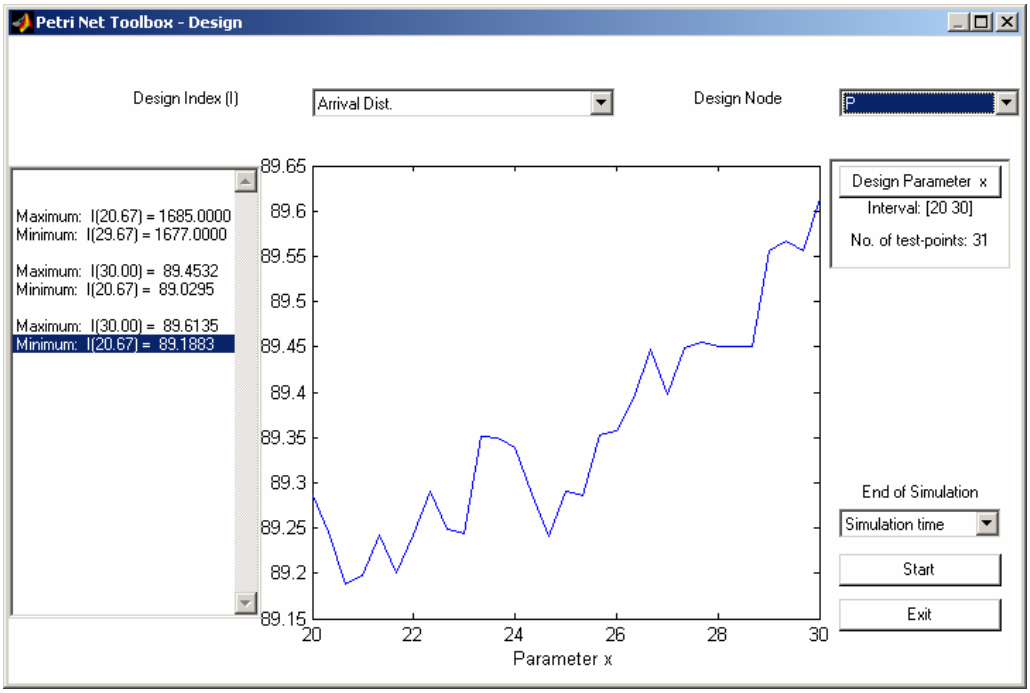


Fig. AP7.4.1. *Evoluția duratei medii de fabricare a unei piese în funcție de* μ.

În cadrul experimentului executat în raport cu parametrul μ, ***Petri Net Toolbox*** memorează toți indicatorii referitori la pozițiile şi tranzițiile modelului analizat. Astfel, informațiile referitoare la durata medie de ocupare a maşinilor M_1, respectiv M_2, pot fi obținute direct, în aceeaşi fereastră, prin selectarea adecvată a indicatorului din câmpurile ***Design Index*** şi ***Design Node***.

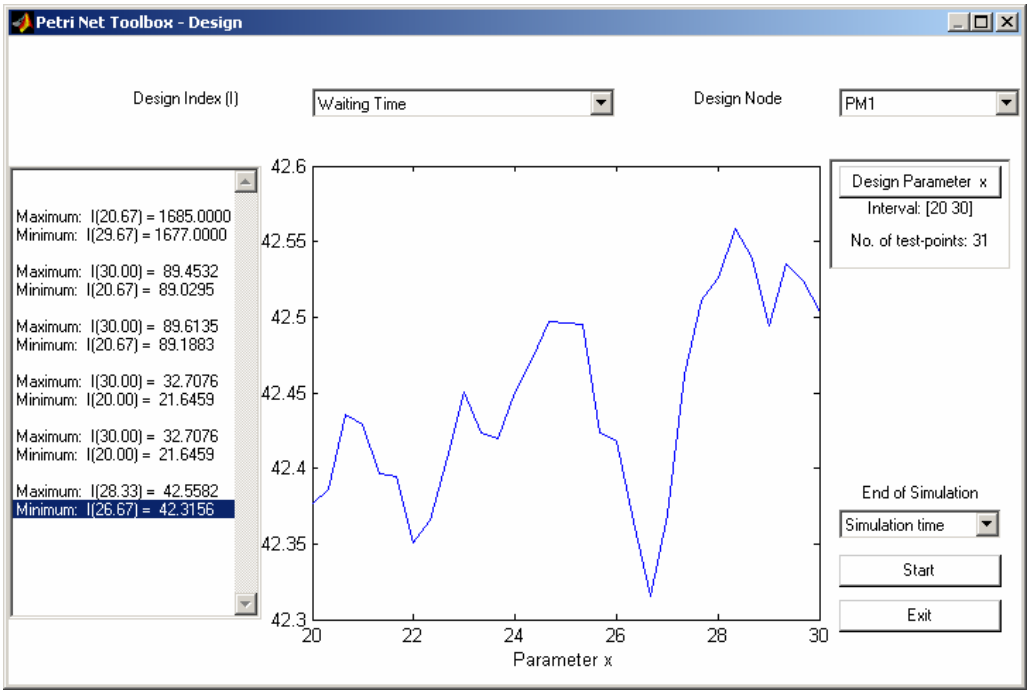


Fig. AP7.4.2. *Evoluția duratei medii de ocupare cu piesă a maşinii* M_1 *în funcție de* μ.

(ii). Pentru dependenţa de μ a duratei medii de ocupare cu piesă a maşinii M_1 se obţine reprezentarea grafică din fig. AP7.4.2. Se constată variaţii foarte mici ale acestui indicator, însemnând (42,56 – 42,38)/42,56= 0,42%.

(iii). Pentru dependenţa de μ a duratei medii de ocupare cu piesă a maşinii M_2 se obţine reprezentarea grafică din fig. AP7.4.3. Se constată variaţii foarte mici ale acestui indicator, însemnând (89,23 – 88,93)/89,23 = 0,33%.

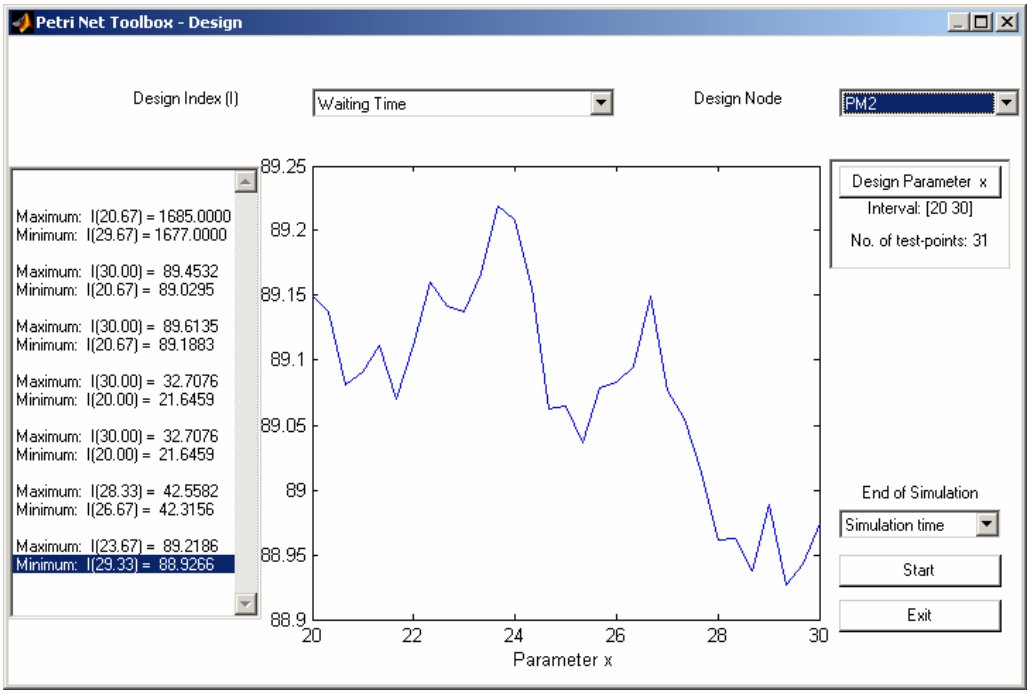


Fig. AP7.4.3. *Evoluţia duratei medii de ocupare cu piesă a maşinii M_2 în funcţie de μ.*

Observaţie: Ca şi în cazul **AP6.5**, remarcăm faptul că durata medie de ocupare cu piesă a maşinii M_1 şi, respectiv M_2, este mai mare decât timpul necesar prelucrării propriu-zise, diferenţa însemnând aşteptarea piesei prelucrate pe maşină pentru efectuarea transportului executat de robotul partajat secvenţial.

AP7.5.

Pentru sistemul de fabricaţie cu reprezentarea schematică din fig. AP3.6.1 şi principiile de funcţionare prezentate în **AP3.6**, se doreşte o exploatare cât mai eficientă din punctul de vedere al duratei medii de fabricaţie a unui produs, în contextul detaliilor de funcţionare prezentate mai jos.

Duratele operaţiilor de prelucrare efectuate asupra unei piese, exprimate în unităţi de timp, au distribuţii uniforme după cum urmează: d_{M1} în intervalul [35, 45] şi d_{M2} în intervalul [80, 90] pentru procesarea pe M_1, respectiv M_2. Duratele operaţiilor de transport, exprimate în unităţi de timp, au valorile $d_{R1} = 20$, $d_{R2} = 20$ pentru descărcarea cu ajutorul lui R a lui M_1, respectiv M_2. Timpul necesitat de eliberarea resurselor M_1, M_2 şi respectiv R este considerat neglijabil.

Se consideră că numărul de palete utilizate pentru fixarea pieselor, notat x, este ajustabil între valorile 2 şi 5. Evitarea apariției deadlock-ului se realizează prin utilizarea unei structuri de control de tip *kanban* (vezi BT4.2.2.4), care să monitorizeze numărul de clienți acceptați pentru toate activitățile dintre tranzițiile ce modelează începerea operațiilor deservite de resursa partajată secvențial (operațiile de transport desfăşurate de robot).

Timpul necesitat de eliberarea unei palete, notat d_P, este considerat constant şi poate fi ales convenabil în intervalul [15, 40] [unități de timp].

1. În funcție de valorile parametrilor x şi d_P, să se reprezinte grafic următorii indicatori:
   (i) durata medie necesară fabricării unui produs finit;
   (ii) durata medie de ocupare cu piesă a maşinii M_1;
   (iii) durata medie de ocupare cu piesă a maşinii M_2;
   (iv) numărul mediu de piese din depozit.
2. Să se precizeze modul de alegere a parametrilor x şi d_P pentru care se poate obține o valoare minimă pentru durata medie de fabricație a unui produs.

Soluție

1. Modelul utilizat pentru rezolvarea problemei este prezentat în fig. AP7.5.1, în care poziția K a fost introdusă pentru a modela activitatea kanban-ului.

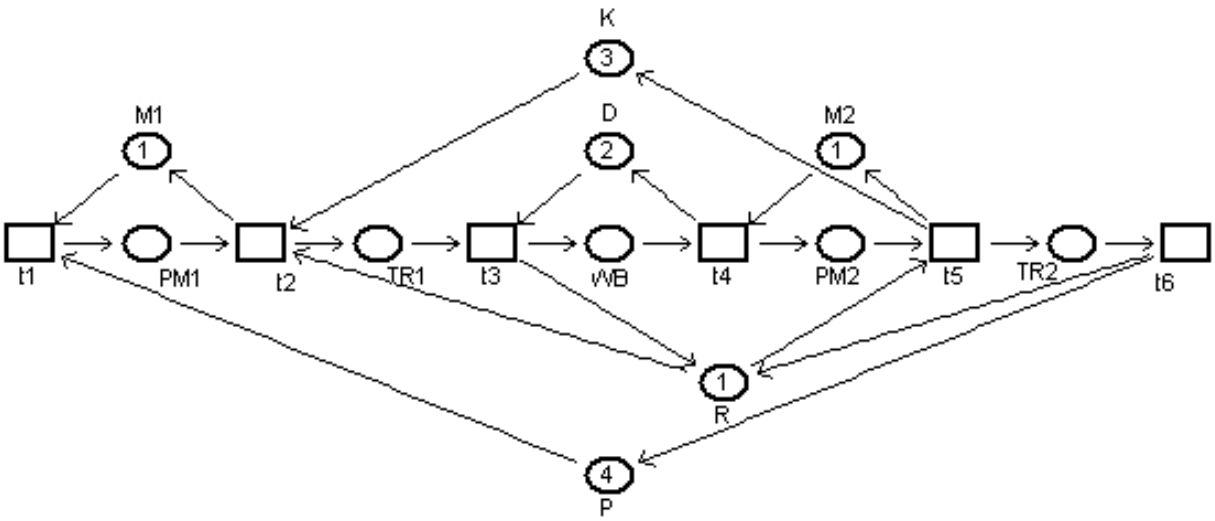


Fig. AP7.5.1. *Modelul utilizat în AP7.5.*

Utilizând facilitatea ***Design*** oferită de mediul ***Petri Net Toolbox*** şi considerând drept parametri numărul x de palete disponibile în sisten (care ia valorile discrete 2, 3, 4 şi 5) şi durata d_P de eliberarea unei palete (luând valori în intervalul [15, 40]), caracteristicile de funcționare investigate pot fi pot fi reprezentate grafic în aceeaşi fereastră ca suprafețe, prin selectarea adecvată a indicatorului din câmpurile ***Design Index*** şi ***Design Node***. Experimentele de simulare utilzate pentru construirea punctelor reprezentative de pe aceste suprafețe au fost efectuate pe durata de 50.000 [unități de timp].

(i). Pentru durata medie necesară fabricării unui produs finit se obţine reprezentarea grafică din fig. AP7.5.2.

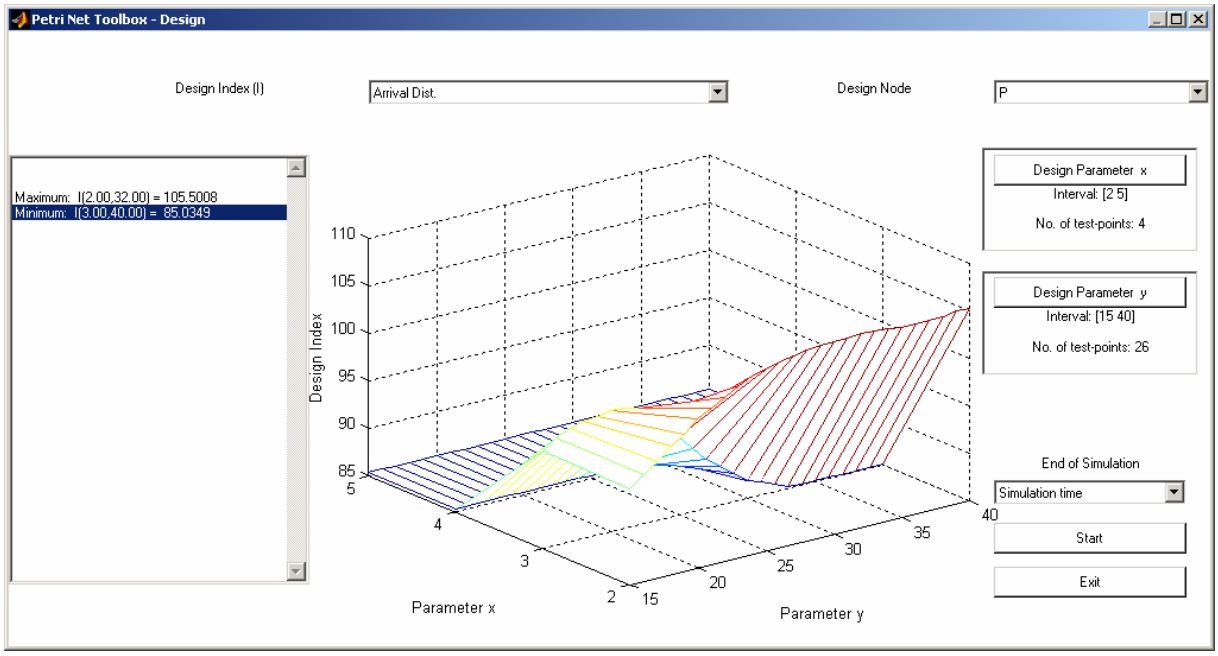


Fig. AP7.5.2. *Reprezentarea grafică a duratei medii necesare fabricării unui produs finit.*

(ii). Pentru durata medie de ocupare cu piesă a maşinii M_1 se obţine reprezentarea grafică din fig. AP7.5.3.

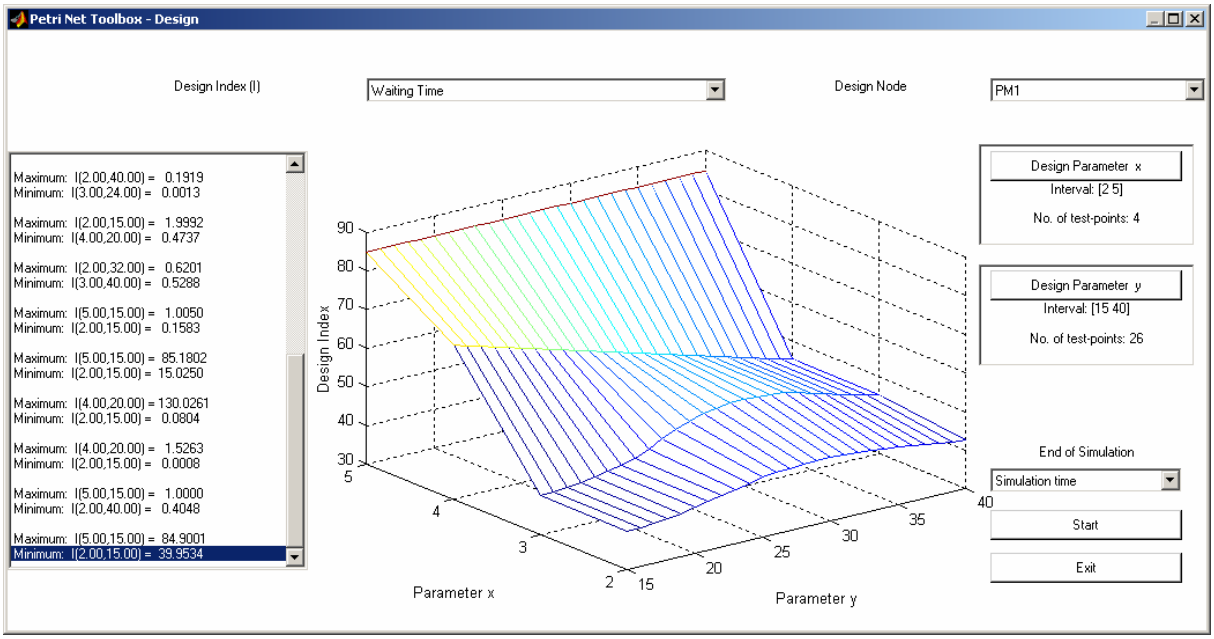


Fig. AP7.5.3. *Reprezentarea grafică a duratei medii de ocupare cu piesă a maşinii M_1.*

(iii). Pentru durata medie de ocupare cu piesă a maşinii M_2 se obține reprezentarea grafică din fig. AP7.5.4.

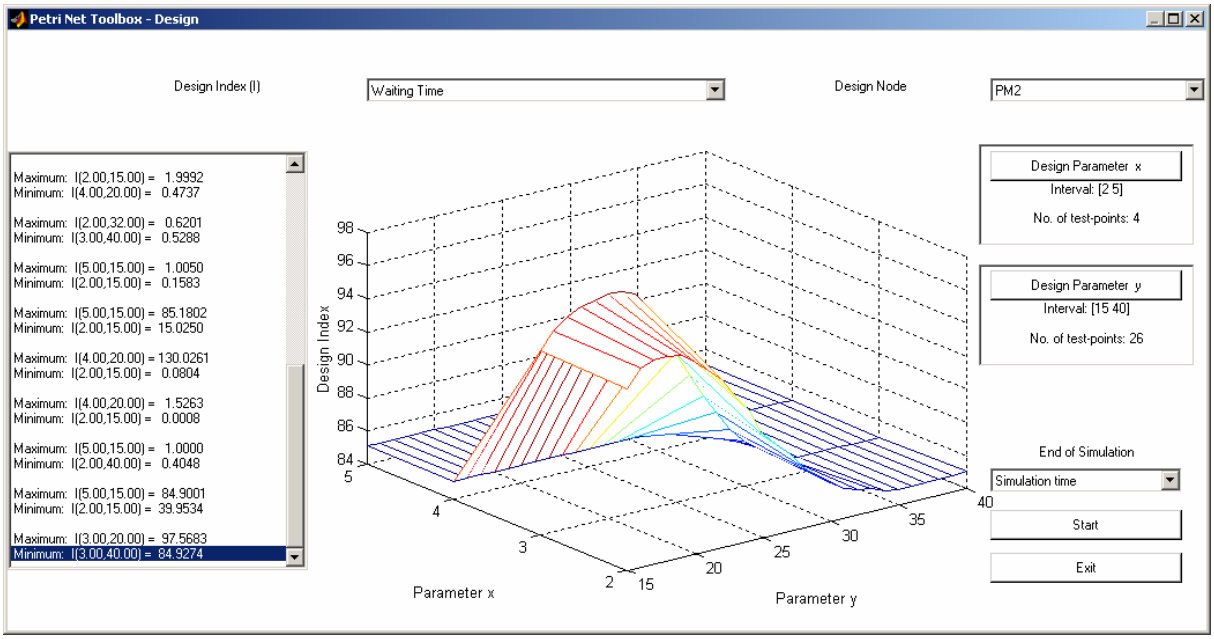


Fig. AP7.5.4. *Reprezentarea grafică a duratei medii de ocupare cu piesă a maşinii M_2.*

(iv). Pentru numărul mediu de piese din depozit se obține reprezentarea grafică din fig. AP7.5.5.

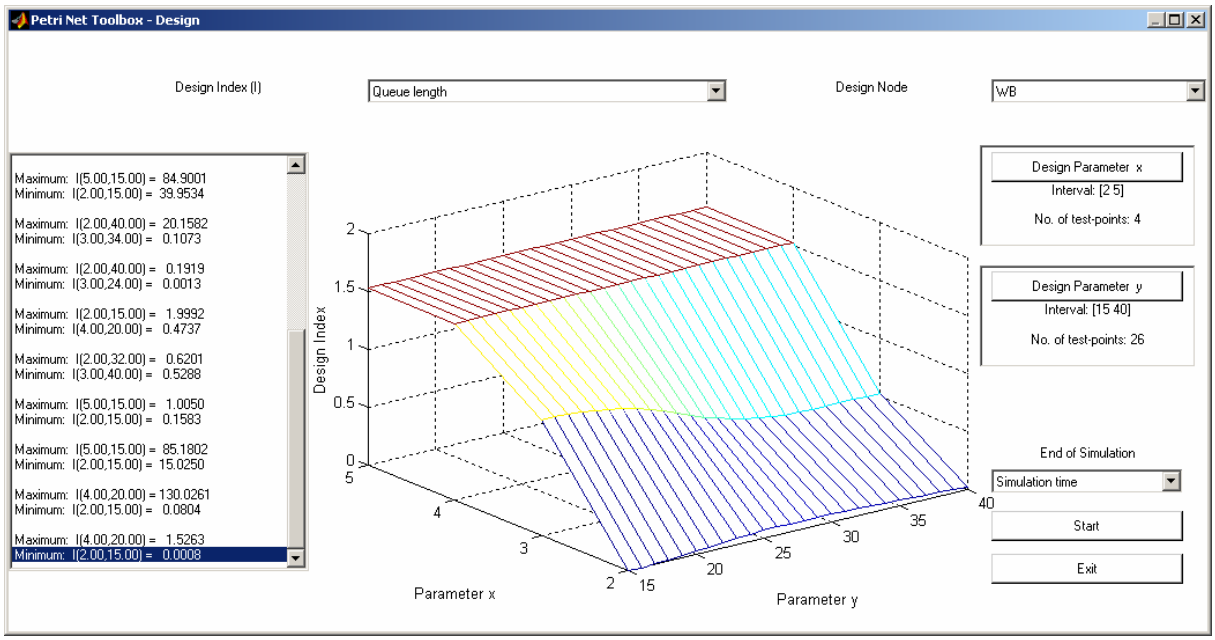


Fig. AP7.5.5. *Reprezentarea grafică a numărului mediu de piese din depozit.*

2. Din examinarea figurii AP7.5.2 rezultă că minimizarea duratei de fabricație a unui produs se pate realiza operând cu 3, 4 sau 5 palete. Dacă utilizăm 3 palete, valoarea minimă a duratei de fabricație este 85 [unități de timp] care se obține pentru durata de eliberare a unei palete $d_p \geq 35$ [unități de timp]. Pentru aceste valori cititorul este invitat să comenteze informațiile furnizate de fig. AP7.5.3 – AP7.5.5. În mod similar se tratează cazurile când în sistem sunt utilizate 4 sau 5 palete.

Capitolul 8

Modele de tip rețea Petri stohastică

Breviar teoretic

BT8.1. Rețele Petri stohastice

O rețea Petri se numește *stohastică* (eng. *stochastic Petri net – SPN*) dacă fiecărei tranziții t_i îi este asociată o variabilă aleatoare X_i de distribuție exponențială, care exprimă întârzierea din momentul validării până la executarea tranziției t_i. Spre deosebire de rețelele Petri cu temporizare *T* stohastică, dacă la un moment dat într-o rețea Petri stohastică mai multe tranziții sunt simultan validate, se va executa mai întâi acea tranziție care posedă întârzierea cea mai scurtă. Astfel, într-o SPN, nu are sens asignarea de probabilități sau priorități de executare pentru tranzițiile aflate în conflict.

Ținând cont de modul de funcționare al rețelelor Petri stohastice prezentat mai sus, într-o SPN, spre deosebire de o rețea Petri cu temporizare *T*, nu există marcaje rezervate. Problema alegerii următoarei tranziții care se va executa se pune de fiecare dată când marcajul rețelei stohastice se modifică. Conflictele se rezolvă în mod natural, prin generarea duratelor corespunzătoare tuturor tranzițiilor validate la momentul respectiv şi selectarea acelei tranziții căreia îi corespunde cea mai mică durată.

Numai în cazul inexistenței conflictelor efective, se obține o echivalență comportamentală între rețelele Petri stohastică şi între rețele Petri cu temporizare *T* stohastică. Cu alte cuvinte, diferențele care apar între dinamicele celor două tipuri de rețele provin tocmai din modalitățile de rezolvare a conflictelor ce pot apărea pentru rețele cu temporizare *T* stohastică (ilustrare în **AP8.1**).

Într-o rețea Petri stohastică rata de executare a unei tranziții *t* poate fi *dependentă de marcaj* în sensul următor. Dacă rata distribuției exponențiale asociată tranziției *t* este λ, atunci la un moment oarecare la care tranziția respectivă este *q*-validată, durata de executare a lui *t* se generează după o distribuție exponențială de rată $q\lambda$. Acest mecanism se utilizează, de exemplu, atunci când numărul de jetoane din pozițiile de intrare a lui *t* (adică ordinul de validare al lui *t*) reprezintă numărul de servere identice în paralel aflate în curs de servire a clienților, în ipoteza că duratele de servire pe toate serverele au aceeaşi distribuție de rată λ (eng. *single-server versus multiple-server semantics*).

Evoluția marcajului unei rețele Petri stohastice reprezintă un *proces Markov omogen*, astfel încât oricărei rețele de acest tip îi poate fi asociat un *lanț Markov omogen*. Pentru

studiul unei SPN, metodele de analiză a lanţurilor Markov (cunoscute din teoria sistemelor de aşteptare) complementează metodele generale de analiză a reţelelor Petri. Rezultatele bazate pe invarianţii de tip *P* sau *T* (prezentate pentru reţelele cu temporizare *P* sau *T*) pot fi extinse la cazul reţelelor Petri stohastice, pentru care se poate vorbi despre conservarea marcajului mediu şi despre frecvenţa medie de executare a tranziţiilor în regim permanent.

BT8.2. Proprietăţi ale legii de distribuţie exponenţială

Legea de *distribuţie exponenţială cu parametrul* $\lambda > 0$ caracterizează o variabilă aleatoare *X* (ce poate lua numai valori reale nenegative) a cărei funcţie de repartiţie este de forma:

$$F(t) = P[X \le t] = \begin{cases} 0 & , \text{ pentru } t < 0, \\ 1 - \mathrm{e}^{-\lambda t}, & \text{ pentru } t \ge 0, \end{cases} \qquad \text{(BT8.2.1)}$$

densitatea de repartiţie fiind

$$f(t) = \begin{cases} 0 & , \text{ pentru } t < 0, \\ \lambda\, \mathrm{e}^{-\lambda t}, & \text{ pentru } t \ge 0. \end{cases} \qquad \text{(BT8.2.2)}$$

Deoarece valoarea medie a variabilei aleatoare este $\mathrm{M}[X] = 1/\lambda$, parametrul λ poartă denumirea de *rată*. Dispersia variabilei aleatoare *X* este $\mathrm{Var}[X] = 1/\lambda^2$.

Importanţa distribuţiei exponenţiale constă în faptul că aceasta este singura distribuţie continuă de probabilitate care posedă proprietatea de „*lipsă de memorie*” (eng. *memoryless property*) exprimată matematic prin relaţia

$$P[X \le s + t \mid X > s] = P[X \le t], \quad \forall t, s \ge 0, \qquad \text{(BT8.2.3)}$$

care rezultă din

$$P[X \le s + t \mid X > s] = 1 - P[X > s + t \mid X > s] = 1 - \frac{P[X > s + t, X > s]}{P[X > s]} =$$

$$= 1 - \frac{P[X > s + t]}{P[X > s]} = 1 - \frac{\mathrm{e}^{-\lambda(t+s)}}{\mathrm{e}^{-\lambda s}} = 1 - \mathrm{e}^{-\lambda t} = P[X \le t], \quad \forall t, s \ge 0.$$

Această proprietate are următoarea interpretare: fie *X* o variabilă aleatoare distribuită exponenţial care reprezintă, de exemplu, timpul de servire a unui client de către un server. Presupunem că servirea clientului începe la momentul $t_0 = 0$. Dacă la momentul $s > 0$ nu a fost terminată servirea clientului, atunci probabilitatea ca servirea clientului să fie terminată până la momentul $s + t$ este egală cu probabilitatea ca timpul total de servire sa fie mai mic decât *t*. Altfel spus, timpul de servire rezidual are distribuţie exponenţială cu aceeaşi rată ca şi timpul de servire total.

O altă proprietate fundamentală a distribuţiei exponenţiale este reprezentată de relaţia cu distribuţia discretă Poisson. Dacă duratele dintre două apariţii consecutive ale unui acelaşi tip de eveniment sunt variabile independente cu distribuţie exponenţială de rată λ, atunci numărul de evenimente care apar într-un intr-un interval de timp [0, *t*), $t > 0$, are distribuţie Poisson de parametru $\alpha = \lambda t$, şi reciproc.

La baza modului de executare a unei tranziţii *t* a cărei rată este *dependentă de marcaj* stă proprietatea că, dacă $X_1, X_2, ..., X_n$ sunt variabile aleatoare independente distribuite

exponențial cu aceeaşi rată λ, atunci variabila $\min\{X_1, X_2, ..., X_n\}$ este distribuită exponențial cu rata $n\lambda$, deoarece

$$\begin{aligned} P[\min\{X_1, X_2, ..., X_n\} \le t] &= 1 - P[\min\{X_1, X_2, ..., X_n\} > t] = \\ &= 1 - P[X_1 > t]P[X_2 > t]...P[X_n > t] = 1 - e^{-n\lambda t}, \quad \forall t \ge 0. \end{aligned} \tag{BT8.2.4}$$

Distribuția exponențială este utilizată pentru a modela procese care „*nu au memorie*", ca de exemplu duratele dintre sosirile clienților într-un sistem (eng. *interarrival times*), timpi de aşteptare (dacă probabilitatea de a mai aştepta încă un interval de timp este independentă de timpul de aşteptare scurs deja), duratele convorbirilor telefonice, timpul de viață al componentelor electronice.

BT8.3. Lanțuri Markov omogene, continue în timp

BT8.3.1. Concepte de bază

Procesele Markov reprezintă un tip particular de procese stohastice, a căror proprietate caracteristică este aceea că nu au memorie. Evoluția viitoare a unui proces Markov este influențată numai de starea curentă a acestuia. Dacă procesul Markov poate avea numai un număr finit sau numărabil de stări, atunci este denumit *lanț Markov* (eng. *Markov chain*). Lanțurile Markov constituie cel mai important exemplu de proces stohastic, studiul matematic al proceselor stohastice putând fi privit ca o generalizare într-un sens sau altul a teoriei lanțurilor Markov.

Un *lanț Markov în timp continuu* (eng. *continuous-time Markov chain* – CTMC) reprezintă un proces stohastic X având o mulțime finită ($\mathcal{S} = \{1, 2, ..., n\} \subset \mathbb{N}^*$) sau numărabilă de stări ($\mathcal{S} = \mathbb{N}^*$). Starea acestui proces este observată la momentele de timp (continuu) $t \in \mathbb{R}_+ = [0, +\infty)$, starea la momentul inițial $t_0 = 0$ fiind $X(0)$. Pentru lanțurile Markov în timp continuu proprietatea caracteristică (de lipsă a memoriei) este precizată prin condiția

$$\begin{aligned} &\forall r, s, t \in \mathbb{R}_+, \ 0 \le r \le s \le t, \\ &P[X(t) = j \,|\, X(s) = i \text{ şi } X(r) = k] = P[X(t) = j \,|\, X(s) = i] \overset{not}{=} p_{ij}(s,t), \quad \forall i, j, k \in \mathcal{S}. \end{aligned} \tag{BT8.3.1}$$

Funcțiile $p_{ij}(s,t)$ definite prin relația precedentă se numesc *funcții de tranziție de stare* (eng: *state transition functions*).

În continuare ne limităm la cazul lanțurilor Markov *omogene*, pentru care funcțiile de tranziție de stare depind numai de diferența dintre cele două momente de timp:

$$P[X(s+t) = j \,|\, X(s) = i] = P[X(t) = j \,|\, X(0) = i] \overset{not}{=} p_{ij}(t), \quad \forall s, t \ge 0, \quad \forall i, j \in \mathcal{S}$$

şi au proprietățile:

$$\begin{aligned} &0 \le p_{ij}(t) \le 1, \quad \forall t \ge 0, \quad \forall i, j \in \mathcal{S}, \quad p_{ij}(0) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \ne j \end{cases} \\ &\sum_{j \in \mathcal{S}} p_{ij}(t) = 1, \quad \forall t \ge 0, \quad \forall i \in \mathcal{S}. \end{aligned} \tag{BT8.3.2}$$

Funcţiile de tranziţie de stare satisfac *ecuaţia Chapman-Kolmogorov* (fig. BT8.3.1):

$$p_{ij}(s+t) = \sum_{r \in \mathcal{S}} p_{ir}(s)\, p_{rj}(t), \quad \forall s,t \geq 0,\ \forall i,j \in \mathcal{S}. \tag{BT8.3.3}$$

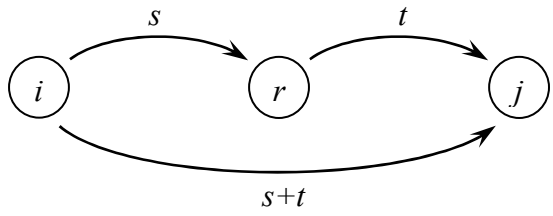


Fig. BT8.3.1.

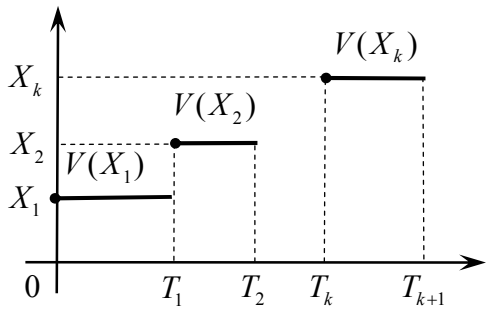


Fig. BT8.3.2.

Matricea funcţiilor de tranziţie de stare, definită prin $\boldsymbol{P}(t) = \left[p_{ij}(t) \right]$, $i,j = 1,2,\ldots$, are proprietatea că suma elementelor de pe fiecare linie este egală cu 1, $\sum_{j \in \mathcal{S}} p_{ij}(t) = 1$, $\forall t \geq 0$, şi:

$$\boldsymbol{P}(0) = \boldsymbol{I}, \qquad \boldsymbol{P}(s+t) = \boldsymbol{P}(s) \cdot \boldsymbol{P}(t), \quad \forall s,t \geq 0.$$

Funcţiile de tranziţie de stare sunt rareori utilizate în mod direct. De regulă, un lanţ Markov în timp continuu este caracterizat prin *ratele tranziţiilor de stare* (eng: *state transition rates*). Matricea ratelor tranziţiilor de stare, notată $\boldsymbol{Q} = \left[q_{ij} \right]$, $i,j = 1,2,\ldots$, este definită prin

$$\boldsymbol{Q} = \left. \frac{d\boldsymbol{P}(t)}{dt} \right|_{t=0} \quad \Leftrightarrow \quad q_{ij} = \left. \frac{dp_{ij}(t)}{dt} \right|_{t=0}, \tag{BT8.3.4}$$

şi are proprietatea suma elementelor de pe fiecare linie este egală cu 0, $\sum_{j \in \mathcal{S}} q_{ij} = 0$.

Din ecuaţia Chapman-Kolmogorov (BT8.3.3) rezultă că matricea funcţiilor de tranziţie de stare satisface problema Cauchy

$$\begin{cases} \dfrac{d\boldsymbol{P}(t)}{dt} = \boldsymbol{P}(t) \cdot \boldsymbol{Q}, & t \geq 0 \\ \boldsymbol{P}(0) = \boldsymbol{I} \end{cases} \tag{BT8.3.5}$$

care admite soluţie unică

$$\boldsymbol{P}(t) = \exp(\boldsymbol{Q}t), \quad t \geq 0, \tag{BT8.3.6}$$

unde $\exp(\boldsymbol{Q}t)$ notează exponenţiala matricei $\boldsymbol{Q}t$, fiind definită prin:

$$\exp(\boldsymbol{Q}t) \overset{def}{=} \sum_{k=0}^{\infty} \frac{1}{k!} \boldsymbol{Q}^k t^k = \boldsymbol{I} + \frac{1}{1!}\boldsymbol{Q}t + \frac{1}{2!}\boldsymbol{Q}^2 t^2 + \ldots. \tag{BT8.3.7}$$

Pentru orice stare $i \in \mathcal{S}$ timpul cât procesul stohastic rămâne în starea i după ce a ajuns în această stare, notat $V(i)$, se numeşte *timp de ocupare a stării* (eng: *state holding time*) (fig. BT8.3.2) şi este o variabilă aleatoare continuă cu distribuţie exponenţială de rată $\Lambda(i) = -q_{ii} = \sum_{j \in \mathcal{S},\, j \neq i} q_{ij} > 0$,

$$P[V(i) \leq t] = 1 - \mathrm{e}^{-\Lambda(i)t}, \quad \forall t \geq 0. \tag{BT8.3.8}$$

Rezultă că timpul mediu de ocupare a stării i este $\mathrm{M}[V(i)] = 1/\boldsymbol{\Lambda}(i)$. De asemenea, timpul de ocupare a stării i, $V(i)$ este independent de starea **în** care va trece procesul la tranziția **din** starea i (nu depinde de următoarea stare vizitată).

BT8.3.2. Analiza tranzițiilor de stare a unui lanț Markov continuu şi omogen

Unul din obiectivele principale în analiza unui lanț Markov în timp continuu este determinarea probabilității ca lanțul să se găsească într-o anumită stare la un moment dat. Se definesc *probabilitățile de stare*:

$$\pi_i(t) = P[X(t) = i], \quad t \geq 0, \quad \forall i \in \mathcal{S}, \tag{BT8.3.9}$$

cu ajutorul cărora se formează vectorul linie $\boldsymbol{\pi}(t) = [\pi_1(t),\ \pi_2(t), \ldots]$, $t \geq 0$. Distribuția de probabilitate a stării inițiale a lanțului este $\boldsymbol{\pi}(0) = [\pi_1(0),\ \pi_2(0), \ldots]$.

Aplicând ecuația Chapman-Kolmogorov se obține

$$\begin{aligned} \pi_j(t) = P[X(t) = j] &= \sum_{i \in \mathcal{S}} P[X(t) = j \,|\, X(0) = i] \cdot P[X(0) = i] = \\ &= \sum_{i \in \mathcal{S}} \pi_i(0) \cdot p_{ij}(t), \quad \forall j \in \mathcal{S}, \end{aligned} \tag{BT8.3.10}$$

relație echivalentă cu

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0) \cdot \boldsymbol{P}(t), \quad t \geq 0, \tag{BT8.3.11}$$

de unde rezultă

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0) \cdot \exp(\boldsymbol{Q}t), \quad t \geq 0. \tag{BT8.3.12}$$

Vectorul probabilităților de stare satisface ecuația diferențială

$$\frac{d\boldsymbol{\pi}(t)}{dt} = \boldsymbol{\pi}(t) \cdot \boldsymbol{Q}, \quad t \geq 0. \tag{BT8.3.13}$$

Probabilitățile staționare de stare (în regimul permanent al sistemului) sunt definite prin

$$\boldsymbol{\pi} = \lim_{t \to \infty} \boldsymbol{\pi}(t) \quad \Leftrightarrow \quad \pi_i = \lim_{t \to \infty} \pi_i(t) \quad , \quad i \in \mathcal{S}, \tag{BT8.3.14}$$

în ipoteza că această limită există. În acest caz, probabilitățile staționare de stare satisfac sistemul de ecuații liniare:

$$\begin{cases} \boldsymbol{\pi} \cdot \boldsymbol{Q} = \boldsymbol{0}, \\ \sum_{i \in \mathcal{S}} \pi_i = 1. \end{cases} \tag{BT8.3.15}$$

BT8.4. Proprietăți ale rețelelor Petri stohastice

BT8.4.1. Construcția lanțului Markov asociat unei rețele Petri stohastice

Lanțurile Markov în timp continuu joacă un rol fundamental în analiza rețelelor Petri stohastice mărginite. Deoarece distribuțiile exponențiale ce caracterizează întârzierile în executarea tranzițiilor nu au memorie, s-a demonstrat că graful de accesibilitate al unei rețele Petri stohastice mărginite este *izomorf* cu un *lanț Markov continuu*.

Lanțul Markov asociat unei rețele Petri stohastice mărginite ($\boldsymbol{N}$, M_0) (pentru care ratele de executare a tranzițiilor sunt dependente sau nu de marcajul pozițiilor predecesor) se obține

din graful de accesibilitate al acesteia în modul următor:

- se consideră mulțimea marcajelor accesibile R(M_0), ca fiind spațiul stărilor lanțului Markov. Numărul s de stări ale lanțului Markov este egal cu cardinalul mulțimii marcajelor accesibile din M_0, R(M_0);
- pentru $i \neq j$, *rata tranziției* din starea M_i în starea M_j este dată de $q_{ij} = \lambda'_k$, unde λ'_k este frecvența de executare a tranziției t_k ce transformă marcajul M_i în marcajul M_j. Ca o generalizare, q_{ij} se definește ca fiind:

$$q_{ij} = \begin{cases} \lambda'_k & \text{, executarea tranziției } t_k \text{ transformă marcajul } M_i \text{ în marcajul } M_j, \\ \lambda'_{k1} + \lambda'_{k2} + ... & \text{, executarea tranzițiilor } t_{k1},\ t_{k2} \text{ ... transformă marcajul } M_i \text{ în marcajul } M_j, \\ 0 & \text{, nu există nici o tranziție care să transforme marcajul } M_i \text{ în marcajul } M_j, \end{cases}$$

- pentru $i = j$, rata q_{ii} se determină din relația:

$$\sum_{j=1}^{s} q_{ij} = 0 \quad \Rightarrow \quad q_{ii} = -\sum_{\substack{j=1 \\ j \neq i}}^{s} q_{ij} \,. \tag{BT8.4.1}$$

Dacă rețeaua Petri stohastică (***N***, M_0) este mărginită şi reversibilă, adică $M_0 \in \mathrm{R}(M),\ \forall M \in \mathrm{R}(M_0)$ (ceea ce implică faptul că graful de accesibilitate al rețelei este tare conex), atunci ea generează un lanț Markov continuu ergodic, pentru care se pot calcula probabilitățile staționare de stare prin rezolvarea sistemului de ecuații liniare:

$$\begin{cases} \boldsymbol{\pi} \cdot \boldsymbol{Q} = 0, \\ \sum_{i=1}^{s} \pi_i = 1, \end{cases} \tag{BT8.4.2}$$

în care $\boldsymbol{Q} = [q_{ij}]$ este matricea ratelor de tranziție (de dimensiune $s \times s$) şi $\boldsymbol{\pi} = [\pi_1\ \pi_2,\ ...\ \pi_s]$ este vectorul probabilităților staționare de stare, π_i fiind probabilitatea ca rețeaua Petri să fie în starea M_i, $i = \overline{1,s}$ (ilustrare în **AP8.1** – **AP8.3**).

BT8.4.2. Evaluarea performanțelor unei rețele Petri stohastice

Utilizând notațiile introduse mai sus se pot evalua *performanțele* unei rețele Petri stohastice (***N***, M_0) mărginite şi reversibile, după cum urmează (Murata, 1989), (David et Alla, 1992) (ilustrare în **AP8.1** – **AP8.3**):

(i) probabilitatea îndeplinirii unei anumite condiții particulare reprezentată printr-o submulțime B de marcaje accesibile din M_0, $B \subset \mathrm{R}(M_0)$, ce au o anumită proprietate:

$$P\{B\} = \sum_{M_i \in B} \pi_i \,; \tag{BT8.4.3}$$

(ii) valoarea medie a numărului de jetoane $\mathrm{M}[M(p_i)]$ dintr-o poziție p_i care este k-mărginită:

$$\mathrm{M}[M(p_i)] = \sum_{n=1}^{k} n\, P\{B(i,n)\}, \tag{BT8.4.4}$$

unde $B(i,n)$ este o submulțime din R(M_0) în care numărul de jetoane din poziția p_i este n;

(iii) numărul mediu de executări ale tranziției t_j în unitatea de timp:

$$f_j = \sum_{M_i \in B_j} \pi_i \lambda'_j , \qquad \text{(BT8.4.5)}$$

unde B_j este setul stărilor (marcajelor) din R(M_0) pentru care tranziția t_j este validată, π_i este probabilitatea ca rețeaua Petri să fie în starea M_i, λ'_j este frecvența de executare a tranziției t_j din marcajul M_i;

(iv) gradul mediu de utilizare a tranziției t_j:

$$u_j = \sum_{M_i \in B_j} \pi_i \left(\frac{\lambda'_j}{-q_{ii}} \right), \qquad \text{(BT8.4.6)}$$

unde B_j este setul stărilor (marcajelor) din R(M_0) pentru care tranziția t_j este validată, π_i este probabilitatea ca rețeaua Petri să fie în starea M_i, λ'_j este frecvența de executare a tranziției t_j din marcajul M_i, $-q_{ii}$ reprezintă suma de frecvențe (rate) de executări ale tranzițiilor validate în starea M_i.

BT8.4.3. Proprietăți de conservare

Se consideră o rețea Petri stohastică pură şi mărginită ($\boldsymbol{N}$, M_0) cu n tranziții şi m poziții, a cărei topologie este descrisă de matricea de incidență $\boldsymbol{A} = [a_{ij}]$ de dimensiune $n \times m$.

Marcajul $M(t)$ al rețelei $\boldsymbol{N}$ la momentul $t \geq 0$ este un vector aleator de dimensiune m; componenta i a acestui vector reprezintă numărul de jetoane din poziția p_i, $i = \overline{1,m}$, la acel moment; se consideră $M_0 = M(0)$. Fie $\boldsymbol{u}(t) = [u_1(t)\ u_2(t) \ldots u_n(t)]^T$ vectorul aleator a cărui componentă $u_j(t)$, $j = \overline{1,n}$, reprezintă numărul (aleator) de executări ale tranziției t_j, $j = \overline{1,n}$, în intervalul de timp $[0,t)$.

În orice moment $t \geq 0$ vectorii $M(t)$ şi $\boldsymbol{u}(t)$ sunt legați prin relația temporală, analogă ecuației de stare (BT3.4.4):

$$M(t) = M_0 + \boldsymbol{A}^T \boldsymbol{u}(t), \quad t \geq 0 . \qquad \text{(BT8.4.7)}$$

Valorile medii ale celor doi vectori aleatori considerați, $\mathrm{M}[M(t)]$ şi $\mathrm{M}[\boldsymbol{u}(t)]$, satisfac relația:

$$\mathrm{M}[M(t)] = M_0 + \boldsymbol{A}^T \mathrm{M}[\boldsymbol{u}(t)], \quad \forall t \geq 0 . \qquad \text{(BT8.4.8)}$$

Dacă vectorul $\boldsymbol{y} \in \mathbb{Z}^m$, $\boldsymbol{y} >\neq \boldsymbol{0}$, este un invariant P al rețelei ($\boldsymbol{A}\boldsymbol{y} = \boldsymbol{0}$), atunci din relația (BT5.2.1) rezultă că vectorul marcajului mediu $\mathrm{M}[M(t)]$ satisface

$$\mathrm{M}[M(t)]^T \boldsymbol{y} = M_0^T \boldsymbol{y} (= \text{constant}), \quad \forall t \geq 0 . \qquad \text{(BT8.4.9)}$$

Considerând că procesul stohastic de marcaj este ergodic şi staționar, adică $M(t)$ converge către aceeaşi limită finită M^* atât în medie temporală cât şi în medie probabilistică, atunci vectorul M^* se numeşte *vector de marcaj mediu în regim permanent* şi verifică relația

$$M^{*T} \boldsymbol{y} = M_0^T \boldsymbol{y} . \qquad \text{(BT8.4.10)}$$

Această relație arată că suma ponderată a marcajelor medii din pozițiile corespunzătoare suportului unui invariant P este egală cu suma ponderată a marcajelor inițiale din acele poziții.

Considerând că procesul stohastic de executare a tranzițiilor este ergodic, adică $\boldsymbol{u}(t)$ converge către aceeaşi limită finită $\boldsymbol{u}^*$ atât în medie temporală cât şi în medie probabilistică, atunci vectorul $\boldsymbol{u}^*$ se numeşte vectorul frecvențelor medii de executare a tranzițiilor în regim permanent şi verifică relația

$$\boldsymbol{A}^T \boldsymbol{u}^* = \boldsymbol{0}\,. \qquad \text{(BT8.4.11)}$$

Scriind relația (BT8.4.11) în forma $(\boldsymbol{A}^+)^T \boldsymbol{u}^* = (\boldsymbol{A}^-)^T \boldsymbol{u}^*$, aceasta exprimă conservarea fluxului de jetoane într-o rețea Petri stohastică mărginită: în regim permanent, fluxul de jetoane care intră în orice poziție este egal cu fluxul de jetoane care iese din acea poziție.

În aceeaşi ipoteză de ergodicitate pentru procesele de marcaj şi de executare a tranzițiilor, *formula lui Little* furnizează timpul mediu de staționare a unui jeton într-o poziție p_i, $i = \overline{1,m}$, notat $\boldsymbol{d}^*(p_i)$, ca raportul dintre marcajul mediu al acelei poziții în regim permanent şi suma ponderată a frecvențelor medii de executare a tranzițiilor de intrare în p_i

$$\boldsymbol{d}^*(p_i) = \frac{M^*(p_i)}{(\boldsymbol{A}_i^+)^T \boldsymbol{u}^*}\,, \qquad \text{(BT8.4.12)}$$

unde $\boldsymbol{A}_i^+$ reprezintă coloana *i* a matricei de incidență de intrare $\boldsymbol{A}^+$.

BT8.5. Rețele Petri stohastice generalizate

Rețelele Petri stohastice generalizate (eng. *generalized stochastic Petri net – GSPN*) au fost introduse pentru a extinde capacitatea de modelare a rețelelor stohastice. Într-o GSPN, în afară de tranziții cu temporizare de tip exponențial există şi tranziții netemporizate (*imediate*). Tranzițiile imediate modelează evenimente a căror apariție are loc instantaneu. În cazul în care două sau mai multe tranziții imediate sunt în conflict, acestora le pot fi asignate probabilitățile de apariție a evenimentelor pe care le modelează. În plus, între tranziții netemporizate se pot impune priorități de executare. O GSPN poate conține, de asemenea, arce inhibitoare având rolul de a inversa logica de validare şi executarea a unor tranziții.

Dacă într-o rețea Petri stohastică generalizată pentru un anumit marcaj sunt validate atât tranziții imediate cât şi tranziții temporizate, se execută numai tranzițiile imediate. Dacă mai multe tranziții imediate sunt în conflict, numai una dintre ele se execută, în conformitate cu probabilitățile sau prioritățile asignate acestora. Un marcaj al unei GSPN pentru care cel puțin o tranziție netemporizată este validată se numeşte *marcaj invizibil* (eng. *vanishing marking*). Marcajele pentru care sunt validate numai tranziții temporizate se numesc *marcaje tangibile* (eng. *tangible markings*).

O rețea stohastică generalizată, ce conține atât marcaje tangibile cât şi marcaje invizibile este, de asemenea, izomorfă cu un lanț Markov continuu care se obține în acelaşi mod ca şi în cazul unei rețele Petri stohastice. Eliminând din acest lanț Markov marcajele invizibile şi tranzițiile netemporizate se obține *lanțul Markov redus* pentru care probabilitățile staționare de stare există dacă rețeaua este mărginită şi reversibilă (ilustrare în **AP8.2**). Aceste condiții restrictive sunt numai suficiente, nu şi necesare, putând fi încălcate în cazul în care modelarea unui sistem sub formă de GSPN se realizează pornind de la lanțul Markov corespunzător şi impunând pentru acesta condițiile de stabilitate (de regim permanent).

Aplicații

AP8.1.

Fie un server care deserveşte două tipuri de clienți în paralel, modelat prin rețeaua Petri stohastică din fig. AP8.1.1. Servirea primului tip de clienți este modelată de subrețeaua din partea stângă a figurii (cu nodurile p_1, t_1, p_2, t_2), iar servirea celui de-al doilea tip de clienți de subrețeaua din partea dreaptă a figurii (cu nodurile p_1, t_3, p_3, t_4). Durata de timp asignată tranziției t_i, $i=\overline{1,4}$, are distribuție exponențială cu rata λ_i [unități de timp $^{-1}$]. Se consideră următoarele valori numerice: $\lambda_1=\lambda_4=1$, $\lambda_2=\lambda_3=10$.

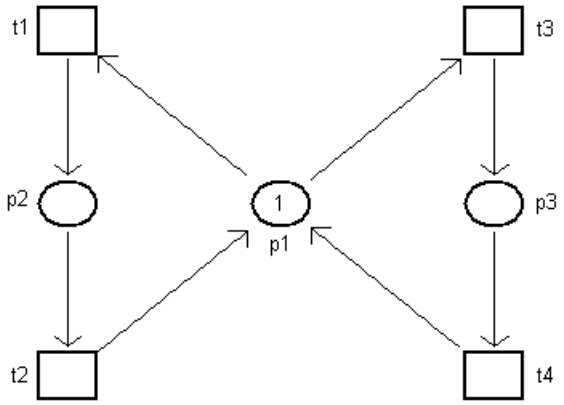


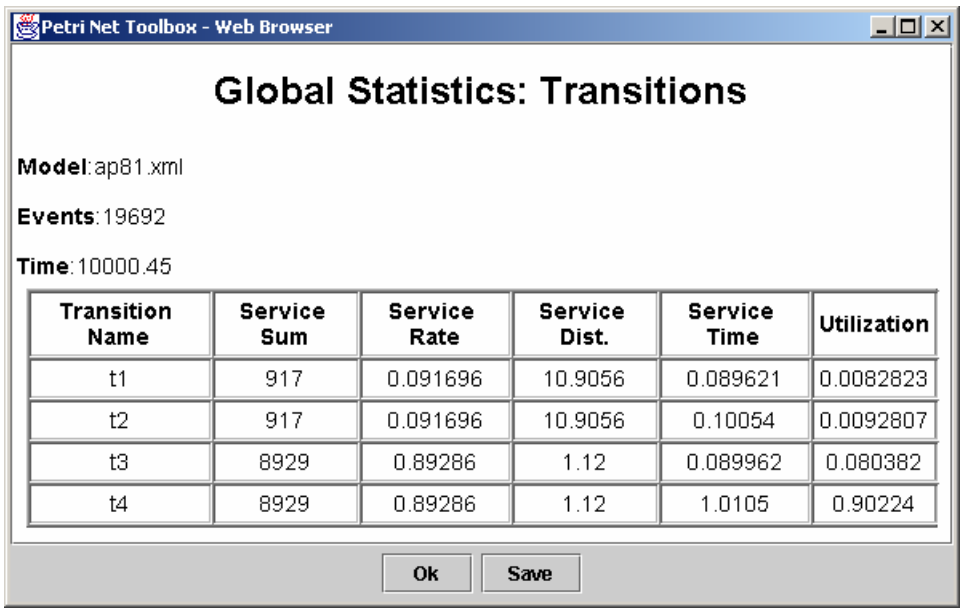
Fig. AP8.1.1. *Topologia rețelei Petri stohastice studiată în AP8.1.*

1. Să se determine, prin simulare în mediul ***Petri Net Toolbox***, numărul de clienți de fiecare tip serviți complet într-un interval de timp de 10.000 [unități de timp].
2. Să se verifice analitic rezultatele obținute la punctul 1.
3. Să se comenteze rezultatele obținute la punctul 1 prin prisma celor obținute la **AP7.1**.

Soluție

1. Rezultatele obținute în urma simulării funcționării sistemului timp de 10.000 [unități de timp] sunt prezentate în fig. AP8.1.2. Numărul de clienți de primul tip serviți complet este dat de indicatorul ***Service Sum*** pentru tranziția t_2 (având valoarea 917), iar numărul de clienți de al doilea tip serviți complet este dat de indicatorul ***Service Sum*** pentru tranziția t_4 (având valoarea 8.929). Se observă că raportul dintre cele două numere este aproximativ egal cu raportul dintre ratele de executare corespunzătoare tranzițiilor t_1 şi t_3 care modelează procesele de sosire în sistem a clienților de primul tip şi, respectiv, de al doilea tip.

2. Pentru rețeaua Petri studiată se construieşte mai întâi arborele de accesibilitate (fig. AP8.1.3.(a)). Aplicând algoritmul prezentat în BT8.4.1, se obține lanțul Markov asociat rețelei (reprezentat în fig. AP8.1.3.(b)) având drept stări cele trei marcaje accesibile din M0.



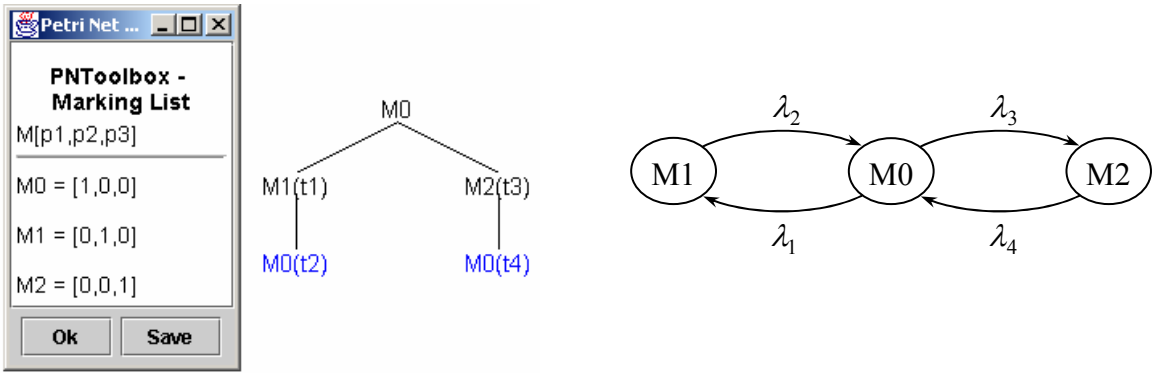
Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	917	0.091696	10.9056	0.089621	0.0082823
t2	917	0.091696	10.9056	0.10054	0.0092807
t3	8929	0.89286	1.12	0.089962	0.080382
t4	8929	0.89286	1.12	1.0105	0.90224

Fig. AP8.1.2. *Indicatorii globali pentru tranzițiile rețelei din fig. AP8.1.1.*

Considerând stările lanțului Markov în ordinea M0, M1, M2, matricea ratelor tranzițiilor de stare este

$$\boldsymbol{Q} = \begin{bmatrix} -\lambda_1 - \lambda_3 & \lambda_1 & \lambda_3 \\ \lambda_2 & -\lambda_2 & 0 \\ \lambda_4 & 0 & -\lambda_4 \end{bmatrix} = \begin{bmatrix} -11 & 1 & 10 \\ 10 & -10 & 0 \\ 1 & 0 & -1 \end{bmatrix}.$$

Vectorul probabilităților staționare de stare, $\boldsymbol{\pi} = [\pi_0 \ \pi_1 \ \pi_2]$, se determină din condițiile $\boldsymbol{\pi} \cdot \boldsymbol{Q} = \boldsymbol{0}$, $\pi_0 + \pi_1 + \pi_2 = 1$, din care rezultă $\pi_0 = 10/111$, $\pi_1 = 1/111$ și $\pi_2 = 100/111$.



(a) (b)

Fig. AP8.1.3. (a) *Arborele de accesibilitate și* (b) *lanțul Markov ce corespund rețelei Petri stohastice din fig. AP8.1.1.*

Deoarece tranziția t_2 se execută numai din marcajul M1, cu rata λ_2, numărul mediu de executări în unitatea de timp este, conform (BT8.4.5), egal cu $f_2 = \pi_1 \lambda_2 = 10/111 = 0,09009$ [unități de timp $^{-1}$], astfel încât în 10.000 [unități de timp] această tranziție se va executa în medie de 900 de ori.

Raționând analog pentru tranziția t4 se obține $f_4 = \pi_2 \lambda_4 = 100/111 = 0,9009$ [unități de timp $^{-1}$], astfel încât în 10.000 [unități de timp] tranziția t4 se va executa în medie de 9000 de ori.

Se observă că rezultatele obținute prin simulare au valori foarte apropiate de valorile calculate analitic.

3. Rețelele din fig. AP7.1.1 şi AP8.1.1 au aceeaşi topologie, acelaşi marcaj inițial şi aceleaşi informații temporale asignate tranzițiilor. Diferențele care apar între dinamicele celor două rețele se datorează modului în care, de fiecare dată când se ajunge în marcajul inițial şi sunt validate t1 şi t3, este selectată următoarea tranziție care va fi executată.

Observație: Rețeaua Petri stohastică din fig. AP8.1.1 poate modela, de exemplu, un server cu posibilități de defectare (vezi **AP2.5**).

AP8.2.

Fie un sistem de fabricație compus dintr-o maşină M şi un depozit D ca în figura de mai jos, capacitatea sistemului (maşină + depozit) fiind limitată la două piese:

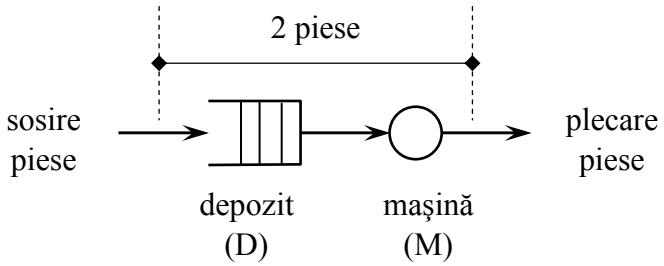


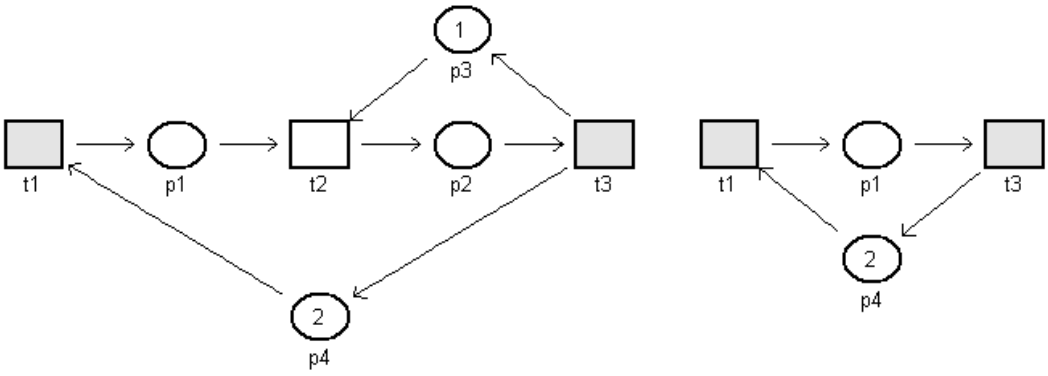
Fig. AP8.2.1. *Reprezentare schematică a sistemului de fabricație studiat în AP8.2.*

Dacă există loc liber în depozit, piesele intră în depozitul D după o distribuție exponențială cu rata λ. Ori de câte ori maşina M este liberă şi există o piesă în depozit, aceasta este preluată imediat pe maşină. Maşina M prelucrează piesele după o distribuție exponențială cu rata μ.

1. Să se construiască modelul de tip rețea Petri stohastică ce descrie funcționarea sistemului, considerând că în starea inițială depozitul este gol şi maşina este neutilizată.
2. Să se construiască lanțul Markov asociat modelului.
3. Considerând $\lambda = 2$ [s^{-1}] şi $\mu = 1$ [s^{-1}], să se determine analitic (în ipoteza că sistemul funcționează în regim permanent):
   (i) probabilitatea cu care sistemul se poate afla în fiecare din stările accesibile;
   (ii) gradul de utilizare a maşinii;
   (iii) frecvența reală de sosire a pieselor în depozit şi frecvența reală de prelucrare a pieselor pe maşina M;
   (iv) numărul mediu de piese ce se pot afla în depozitul D sau se prelucrează de către maşina M;
   (v) timpul mediu petrecut de o piesă în sistem.
4. Să se verifice, prin simulare în mediul ***Petri Net Toolbox***, rezultatele obținute la punctul 3.

Soluţie

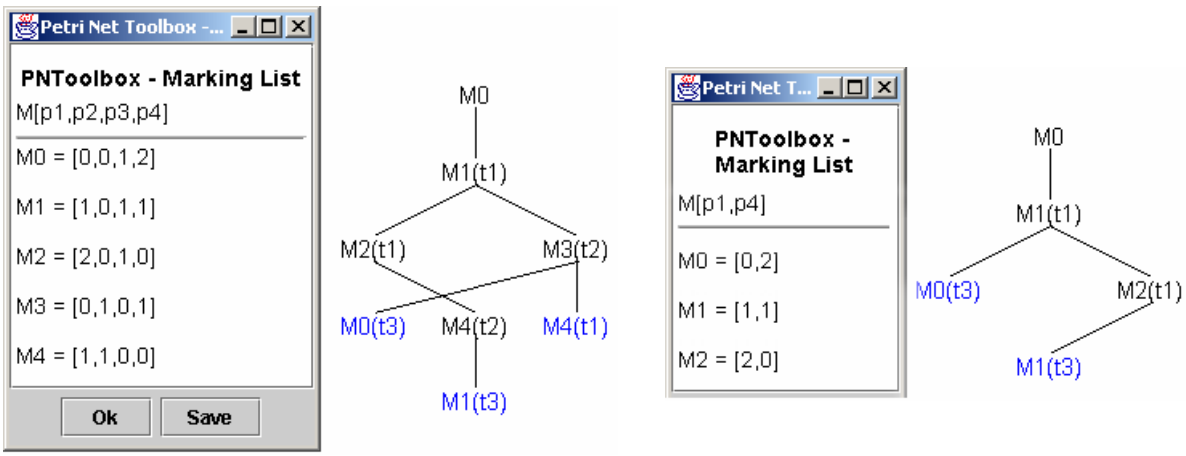
1. Sistemul de fabricaţie studiat poate fi reprezentat prin reţeaua Petri stohastică generalizată din fig. AP8.2.2.(a), în care sunt modelate explicit cele două situaţii în care se poate găsi o piesă (client) în sistem, anume aşteptând în depozit (poziţia p_1) sau în curs de prelucrare (servire) (poziţia p_2). Tranziţiile t_1 şi t_3 modelează procesele de sosire şi, respectiv, de servire a clienţilor, fiind temporizate. Tranziţia t_2, care modelează trecerea unei piese din depozit pe maşină, nu este temporizată. Reţeaua din fig. AP8.2.2.(b), în care ambele tranziţii sunt temporizate, reprezintă un model simplificat al aceluiaşi sistem.



(a) (b)

Fig. AP8.2.2. *Reţelele Petri* (a) *stohastică generalizată şi* (b) *stohastică ce modelează sistemul de fabricaţie din fig. AP8.2.1.*

2. Lanţul Markov asociat fiecăreia dintre reţelele Petri din fig. AP8.2.2 se construieşte pornind de la arborele de accesibilitate corespunzător (fig. AP8.2.3). Pentru reţeaua stohastică generalizată marcajele M1 şi M2 sunt invizibile, tranziţia imediată t_2 fiind validată. Prin eliminarea acestor marcaje, lanţul Markov asociat reţelei va avea trei stări, reprezentate de marcajele M0, M3 şi M4.



(a) (b)

Fig. AP8.2.3. *Arborii de accesibilitate ai reţelelor din* (a) *fig. AP8.2.2.*(*a*) *şi* (b) *fig. AP8.2.2.*(*b*).

Concret, sistemul de fabricație se poate afla în una dintre următoarele trei stări: „0” – nici o piesă în sistem, „1” – o singură piesă în sistem (în curs de prelucrare) şi „2” – două piese în sistem (una în curs de prelucrare şi una în depozit). Tranzițiile din „0” în „1” şi din „1” în „2” au loc cu rata λ (rata de sosire a clienților în sistem), iar tranzițiile din „2” în „1” şi din „1” în „0” au loc cu rata μ (rata de servire). Lanțul Markov corespunzător este prezentat în fig. AP8.2.4, fiind un tip de lanț binecunoscut, denumit *birth-death* (deoarece tranzițiile de stare au loc numai între stări învecinate).

La acelaşi lanț Markov se ajunge şi analizând rețeaua stohastică din fig. AP8.2.2.(b), cele trei stări ale lanțului Markov (cu aceeaşi semnificație fizică de mai sus) corespunzând marcajelor M0, M1 şi M2.

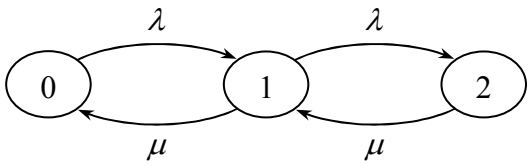


Fig. AP8.2.4. *Lanțul Markov asociat rețelei Petri stohastice din fig. AP8.2.2.*(*b*).

3.(i). Matricea ratelor de tranziție de stare corespunzătoare lanțului Markov din fig. AP8.2.4 este

$$\boldsymbol{Q} = \begin{bmatrix} -\lambda & \lambda & 0 \\ \mu & -(\lambda+\mu) & \lambda \\ 0 & \mu & -\mu \end{bmatrix}.$$

Vectorul probabilităților staționare de stare, $\boldsymbol{\pi} = [\pi_0 \;\; \pi_1 \;\; \pi_2]$, se determină din condițiile $\boldsymbol{\pi} \cdot \boldsymbol{Q} = \boldsymbol{0}$, $\pi_0 + \pi_1 + \pi_2 = 1$, care implică $\pi_1 = \frac{\lambda}{\mu}\pi_0$, $\pi_2 = \frac{\lambda}{\mu}\pi_1$ şi $\pi_0 = \left[1 + \frac{\lambda}{\mu} + \left(\frac{\lambda}{\mu}\right)^2\right]^{-1}$.

Pentru valorile numerice precizate în enunț rezultă $\pi_0 = 4/7$, $\pi_1 = 2/7$ şi $\pi_2 = 1/7$.

(ii). Deoarece maşina (serverul) este utilizată atunci când există cel puțin o piesă (un client) în sistem, gradul de utilizare a maşinii este $\pi_1 + \pi_2 = 1 - \pi_0 = 3/7$.

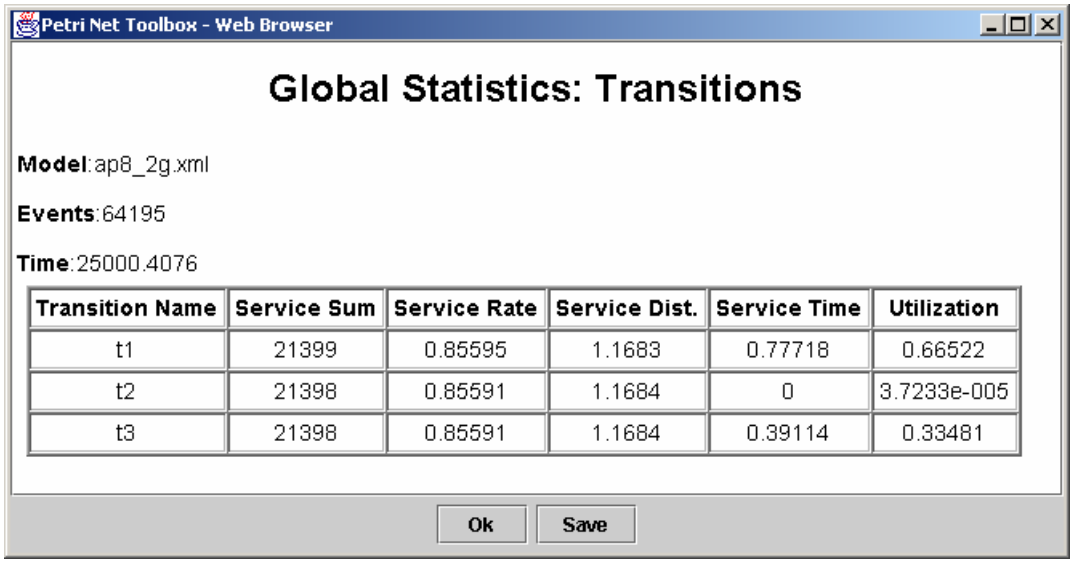
(iii). În sistem pot sosi piese numai atunci când sistemul se află într-o stare diferită de „2”, astfel că frecvența reală de sosire a pieselor în depozit (rata efectivă de sosire a clienților) este dată de $\lambda(1-\pi_2) = 6/7$ [s^{-1}]. Frecvența reală de prelucrare a pieselor pe maşina M (rata de plecare a clienților), fiind dată de $\mu(1-\pi_0) = 6/7$ [s^{-1}], este egală cu frecvența reală de sosire a pieselor.

(iv). Numărul mediu de piese aflate în sistem (în depozitul D sau în curs de prelucrare de către maşina M), notat $M[X]$, se poate calcula, de asemenea, pe baza probabilităților staționare de stare, sub forma $M[X] = \pi_1 + 2\pi_2 = 4/7$.

(v). Timpul mediu petrecut de o piesă în sistem, notat $M[S]$, reprezintă raportul dintre numărul mediu de piese din sistem şi frecvența reală de sosire a pieselor în sistem

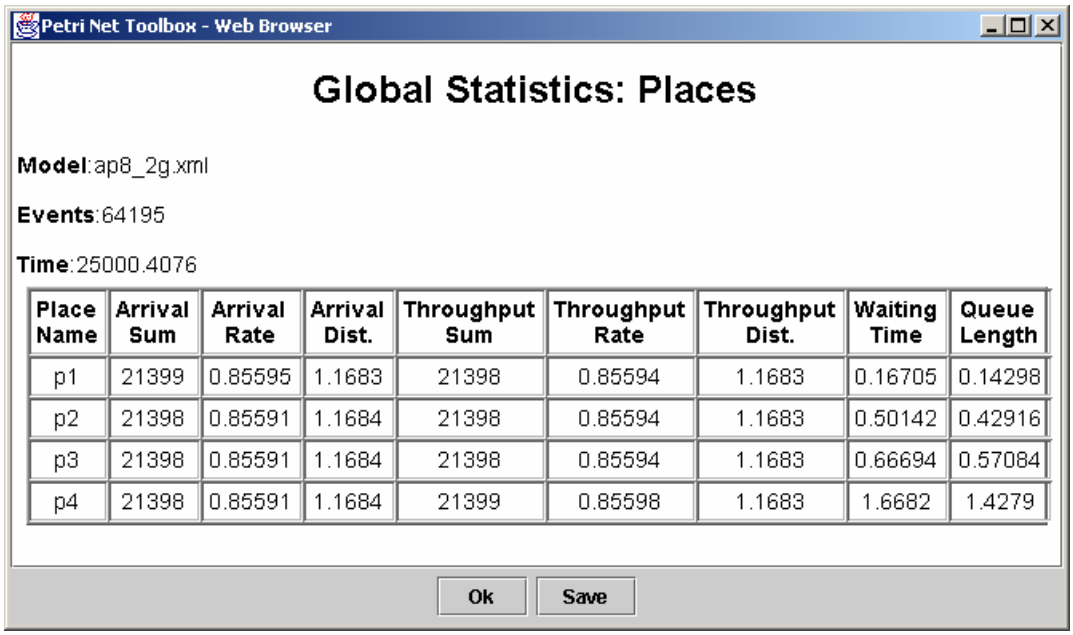
$$M[S] = \frac{M[X]}{\lambda(1-\pi_2)} = \frac{2}{3}.$$

4. Prin simularea în mediul ***Petri Net Toolbox*** a funcționării sistemului de fabricație pe o durată de 25.000 [s] se obțin indicatorii globali prezentați în fig. AP8.2.5 pentru tranzițiile și pozițiile rețelei stohastice generalizate din fig. AP8.2.2.(a). Cititorul este invitat să pună în corespondență rezultatele determinate analitic la punctul 3 cu cele obținute prin simulare.



Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	21399	0.85595	1.1683	0.77718	0.66522
t2	21398	0.85591	1.1684	0	3.7233e-005
t3	21398	0.85591	1.1684	0.39114	0.33481

(a)



Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
p1	21399	0.85595	1.1683	21398	0.85594	1.1683	0.16705	0.14298
p2	21398	0.85591	1.1684	21398	0.85594	1.1683	0.50142	0.42916
p3	21398	0.85591	1.1684	21398	0.85594	1.1683	0.66694	0.57084
p4	21398	0.85591	1.1684	21399	0.85598	1.1683	1.6682	1.4279

(b)

Fig. AP8.2.5. *Indicatorii globali pentru* (a) *tranzițiile și* (b) *pozițiile rețelei din fig. AP8.2.2.*(*a*).

AP8.3.

Se consideră rețeaua Petri stohastică prezentată în fig. AP8.3.1 (Murata, 1989). Tranziția t_2 se execută cu o rată dependentă de marcajul poziției predecesor, adică $m_2\lambda_2$, unde m_2 notează marcajul curent al poziției p_2. Ratele de executare corespunzătoare tranzițiilor t_1, t_3, t_4 şi t_5, notate λ_1, λ_3, λ_4 şi, respectiv, λ_5, sunt independente de marcajul pozițiilor predecesor având următoarele valori numerice: $\lambda_1 = \lambda_5 = 0.5$ [s^{-1}] şi $\lambda_2 = \lambda_3 = \lambda_4 = 1$ [s^{-1}].

1. Să se simuleze dinamica rețelei pe durata corespunzătoare la 30.000 [s] şi să se determine următorii indicatori de performanță:
   (i) numărul mediu de jetoane în unitatea de timp din fiecare poziție a rețelei;
   (ii) numărul mediu de executări în unitatea de timp pentru fiecare tranziție a rețelei;
   (iii) gradul de utilizare a fiecărei tranziții din rețea.
2. Să se verifice prin calcul analitic rezultatele obținute la punctul 1.

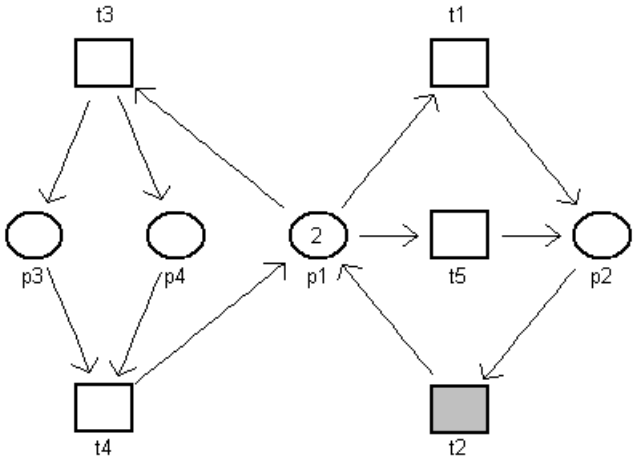


Fig. AP8.3.1. *Rețeaua Petri stohastică studiată în AP8.3.*

Soluție

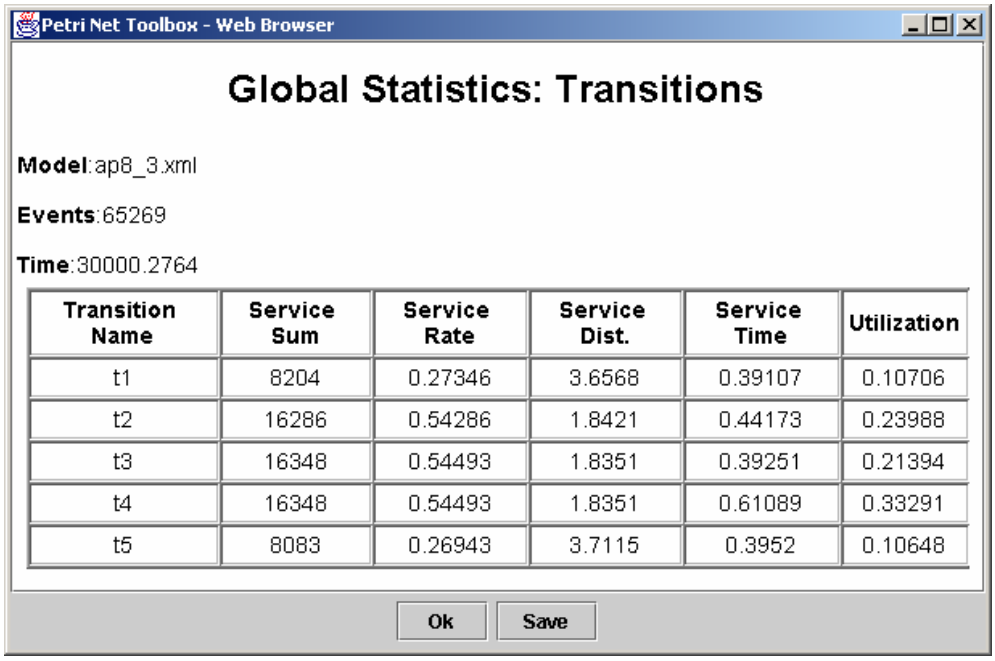
1. Utilizând mediul ***Petri Net Toolbox*** pentru o tranziție temporizată dintr-o rețea Petri stohastică (generalizată), se poate impune ca rata de executare să fie dependentă de marcajul pozițiilor predecesor prin setarea opțiunii ***Marking dependent*** din fereastra de editare corespunzătoare. Această opțiune nu este disponibilă pentru rețelele cu temporizare *T* (deterministă sau stochastică).

Prin simularea dinamicii rețelei pe o durată de 30.000 [s] se obțin indicatorii globali din fig. AP8.3.2.

(i). Numărul mediu de jetoane în unitatea de timp din fiecare poziție a rețelei este precizat de indicatorul ***Queue Length***.

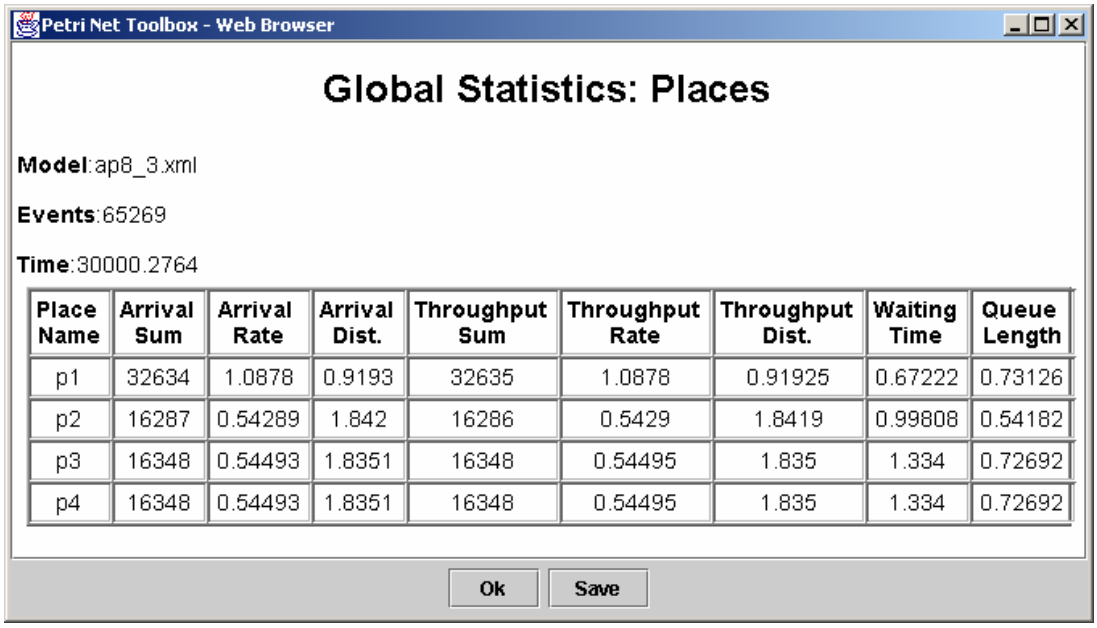
(ii). Numărul mediu de executări în unitatea de timp pentru fiecare tranziție a rețelei este precizat de indicatorul ***Service Rate***.

(iii). Gradul de utilizare al fiecărei tranziții din rețea este precizat de indicatorul ***Utilization***.



Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
t1	8204	0.27346	3.6568	0.39107	0.10706
t2	16286	0.54286	1.8421	0.44173	0.23988
t3	16348	0.54493	1.8351	0.39251	0.21394
t4	16348	0.54493	1.8351	0.61089	0.33291
t5	8083	0.26943	3.7115	0.3952	0.10648

(a)



Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
p1	32634	1.0878	0.9193	32635	1.0878	0.91925	0.67222	0.73126
p2	16287	0.54289	1.842	16286	0.5429	1.8419	0.99808	0.54182
p3	16348	0.54493	1.8351	16348	0.54495	1.835	1.334	0.72692
p4	16348	0.54493	1.8351	16348	0.54495	1.835	1.334	0.72692

(b)

Fig. AP8.3.2. *Indicatorii globali de performanță referitori la* (a) *tranzițiile şi* (b) *pozițiile rețelei Petri din fig. AP8.3.1, corespunzători unei durate de funcționare de 30.000* [*s*].

2. Verificarea analitică a rezultatelor de la punctul 1 se bazează pe analiza lanțului Markov asociat rețelei. Acesta poate fi construit pornind de la arborele de accesibilitate corespunzător (fig. AP8.3.3) aplicând algoritmul prezentat în paragraful BT8.4.1.

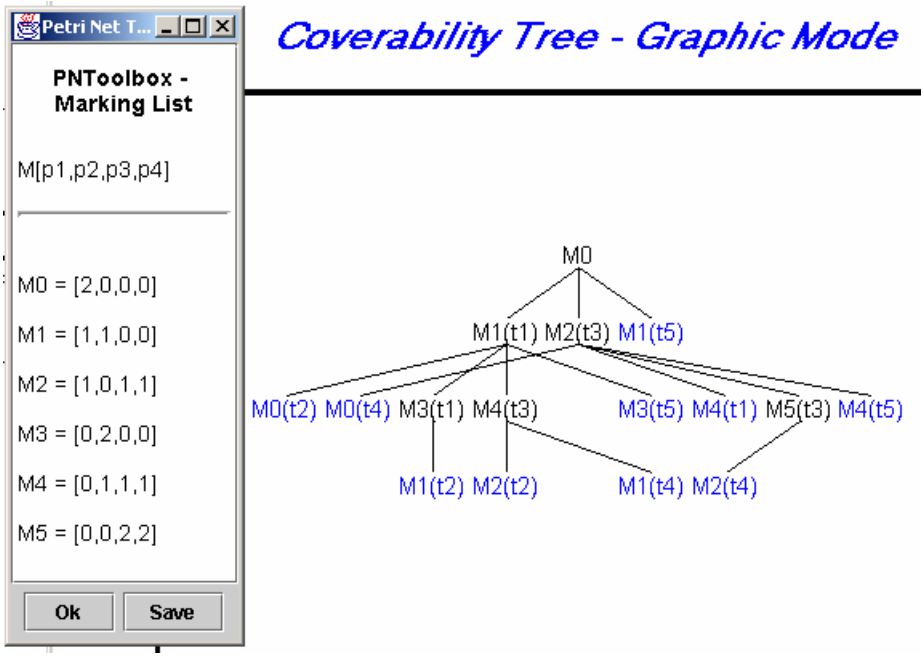


Fig. AP8.3.3. *Arborele de accesibilitate al rețelei Petri stohastice studiate în AP8.3.*

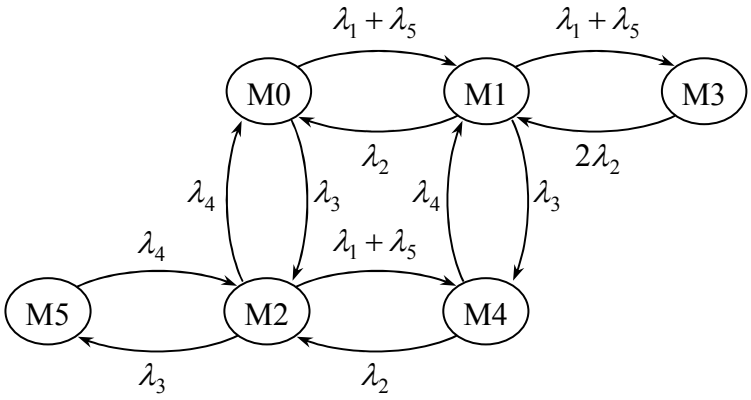


Fig. AP8.3.4. *Lanțul Markov asociat rețelei Petri stohastice studiate în AP8.3.*

Considerând că stările lanțului Markov corespund marcajelor M0, M1, …, M5, din fig AP8.3.3, rezultă fig. AP8.3.4, iar matricea ratelor de tranziție de stare are forma:

$$\boldsymbol{Q}=\begin{bmatrix} -\lambda_1-\lambda_3-\lambda_5 & \lambda_1+\lambda_5 & \lambda_3 & 0 & 0 & 0 \\ \lambda_2 & -\lambda_1-\lambda_2-\lambda_3-\lambda_5 & 0 & \lambda_1+\lambda_5 & \lambda_3 & 0 \\ \lambda_4 & 0 & -\lambda_1-\lambda_3-\lambda_4-\lambda_5 & 0 & \lambda_1+\lambda_5 & \lambda_3 \\ 0 & 2\lambda_2 & 0 & -2\lambda_2 & 0 & 0 \\ 0 & \lambda_4 & \lambda_2 & 0 & -\lambda_2-\lambda_4 & 0 \\ 0 & 0 & \lambda_4 & 0 & 0 & -\lambda_4 \end{bmatrix}.$$

Cu valorile numerice din enunț, se obține:

$$\boldsymbol{Q} = \begin{bmatrix} -2 & 1 & 1 & 0 & 0 & 0 \\ 1 & -3 & 0 & 1 & 1 & 0 \\ 1 & 0 & -3 & 0 & 1 & 1 \\ 0 & 2 & 0 & -2 & 0 & 0 \\ 0 & 1 & 1 & 0 & -2 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix}.$$

Vectorul probabilităților staționare de stare, $\boldsymbol{\pi} = [\pi_0 \ \pi_1 \ \pi_2 \ \pi_3 \ \pi_4 \ \pi_5]$, are drept elemente probabilitățile π_i ca lanțul (în regim permanent) să se găsească în starea $\mathrm{M}i$, $i = 0,\ldots,5$. Rezolvând sistemul de ecuații liniare (BT8.4.2)

$$\begin{cases} \boldsymbol{\pi} \cdot \boldsymbol{Q} = \boldsymbol{0} \\ \pi_0 + \ldots + \pi_5 = 1 \end{cases}$$

se obțin valorile $\pi_0 = \pi_1 = \pi_2 = \pi_4 = \pi_5 = 2/11$ şi $\pi_3 = 1/11$. Valorile indicatorilor de performanță de la punctul 1 pot fi calculate pe baza probabilităților staționare de stare.

(i). Numărul mediu de jetoane în unitatea de timp din fiecare poziție a rețelei se calculează aplicând relația (BT8.4.4). De exemplu, în poziția p_1 se pot afla: 2 jetoane în marcajul M0, 1 jeton în marcajul M1 şi 1 jeton în marcajul M2, astfel că se obține:

$$\mathrm{M}[M(p_1)] = 2\pi_0 + \pi_1 + \pi_2 = \frac{8}{11} = 0,7273.$$

Un raționament similar conduce la

$$\mathrm{M}[M(p_2)] = \pi_1 + 2\pi_3 + \pi_4 = \frac{6}{11} = 0,5455.$$

$$\mathrm{M}[M(p_3)] = \pi_2 + \pi_4 + 2\pi_5 = \frac{8}{11} = 0,7273.$$

$$\mathrm{M}[M(p_4)] = \pi_2 + \pi_4 + 2\pi_5 = \frac{8}{11} = 0,7273.$$

(ii). Numărul mediu de executări în unitatea de timp pentru fiecare tranziție a rețelei se calculează aplicând relația (BT8.4.5). De exemplu, tranziția t_2 se execută cu rata λ_2 din marcajele M1 şi M4, şi cu rata $2\lambda_2$ din marcajul M3, astfel că rata sa medie de executare este:

$$f_2 = \lambda_2\pi_1 + 2\lambda_2\pi_3 + \lambda_2\pi_4 = \frac{6}{11} = 0,5455.$$

Analog se calculează

$$f_1 = \lambda_1\pi_0 + \lambda_1\pi_1 + \lambda_1\pi_2 = \frac{3}{11} = 0,2727,$$

$$f_3 = \lambda_3\pi_0 + \lambda_3\pi_1 + \lambda_3\pi_2 = \frac{6}{11} = 0,5455,$$

$$f_4 = \lambda_4\pi_2 + \lambda_4\pi_4 + \lambda_4\pi_5 = \frac{6}{11} = 0,5455,$$

$$f_5 = \lambda_5\pi_0 + \lambda_5\pi_1 + \lambda_5\pi_2 = \frac{3}{11} = 0,2727.$$

(iii). Gradul de utilizare al fiecărei tranziții din rețea se calculează aplicând relația (BT8.4.6). Cu notațiile $q_{00}=-(\lambda_1+\lambda_3+\lambda_5)$, $q_{11}=-(\lambda_1+\lambda_2+\lambda_3+\lambda_5)$, $q_{22}=-(\lambda_1+\lambda_3+\lambda_4+\lambda_5)$, $q_{33}=-2\lambda_2$, $q_{44}=-(\lambda_2+\lambda_4)$, $q_{55}=-\lambda_4$ pentru elementele diagonalei principale a matricei $\boldsymbol{Q}$ se obține:

$$u_1=\frac{\lambda_1}{-q_{00}}\pi_0+\frac{\lambda_1}{-q_{11}}\pi_1+\frac{\lambda_1}{-q_{22}}\pi_2=\frac{7}{66}=0{,}106,$$
$$u_2=\frac{\lambda_2}{-q_{11}}\pi_1+\frac{2\lambda_2}{-q_{33}}\pi_3+\frac{\lambda_2}{-q_{44}}\pi_4=\frac{8}{33}=0{,}2424,$$
$$u_3=\frac{\lambda_3}{-q_{00}}\pi_0+\frac{\lambda_3}{-q_{11}}\pi_1+\frac{\lambda_3}{-q_{22}}\pi_2=\frac{7}{33}=0{,}2121,$$
$$u_4=\frac{\lambda_4}{-q_{22}}\pi_2+\frac{\lambda_4}{-q_{44}}\pi_4+\frac{\lambda_4}{-q_{55}}\pi_5=\frac{1}{3}=0{,}333,$$
$$u_5=\frac{\lambda_5}{-q_{00}}\pi_0+\frac{\lambda_5}{-q_{11}}\pi_1+\frac{\lambda_5}{-q_{22}}\pi_2=\frac{7}{66}=0{,}106.$$

Rezultatele obținute analitic pot fi comparate cu cele obținute prin simulare în ***Petri Net Toolbox***.

AP8.4.

Se consideră o rețea de aşteptare alcătuită din trei servere şi firelele de aşteptare ce le preced, conectate ca în fig. AP8.4.1 (Cassandras, 1993). Durata de timp dintre sosirile în sistem a doi clienți succesivi are o distribuție exponențială de rată $r_1=1$ [s^{-1}]. Pentru serverul i, $i=\overline{1,3}$, durata de servire a unui client are distribuție exponențială de rată μ_i, $\mu_1=\mu_2=4$ [s^{-1}] şi $\mu_3=2$ [s^{-1}]. Probabilitatea de rutare (dirijare) a unui client de la serverul i către serverul j este notată p_{ij}, $i,j=\overline{1,3}$, şi ia valoarea: $p_{11}=30\%$, $p_{12}=20\%$, $p_{13}=50\%$, $p_{21}=20\%$ şi $p_{22}=80\%$. Probabilitățile p_{ij} ce nu apar în lista precedentă sunt nule şi, în aceste cazuri, nu au loc rutări de la serverul i la serverul j, neavând corespondent grafic în fig. AP8.4.1.

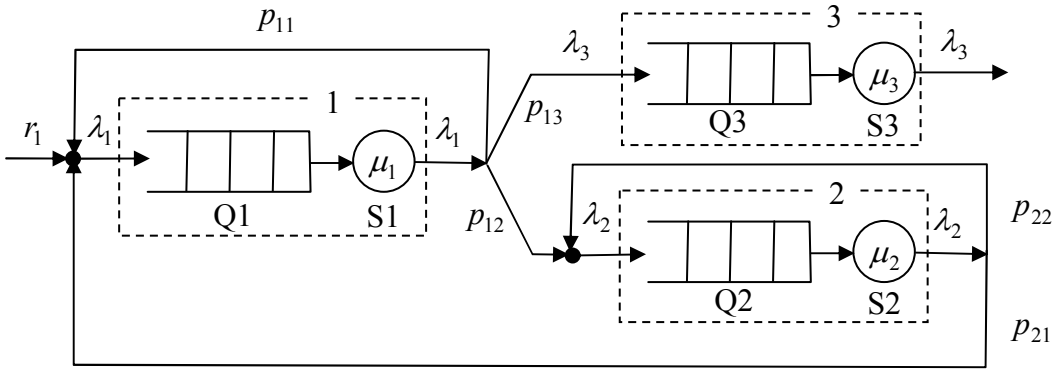


Fig. AP8.4.1. *Reprezentare schematică a rețelei de aşteptare studiată în AP8.4.*

1. Să se construiască o reţea Petri stohastică ce modelează funcţionarea sistemului, considerând că în starea iniţială nu există nici un client în sistem.
2. Pentru fiecare din serverele sistemului, să se determine prin simulare în mediul ***Petri Net Toolbox,*** pe un interval de timp de 10.000 [s], următoarele performanţe:
   (i) rata de sosire a clienţilor;
   (ii) gradul de utilizare;
   (iii) numărul mediu de clienţi din firul de aşteptare ce îl precede şi durata medie de aşteptare per client.

Soluţie

1. Reţeaua de aşteptare studiată este modelată prin reţeaua Petri stohastică generalizată din fig. AP8.4.2 în care tranziţia In modelează procesul de sosire a clienţilor în reţea, fiind temporizată cu rata r_1. Tranziţiile m1, m2 şi m2 modelează procesele de servire a clienţilor de către cele trei servere, fiind temporizate cu ratele μ_1, μ_2 şi, respectiv, μ_3. Procesul de rutare a clienţilor serviţi de primul server este modelat de tranziţiile netemporizate t11, t12 şi t13, cărora le sunt asignate probabilităţile p_{11}, p_{12} şi, respectiv, p_{13}. Similar, rutarea clienţilor serviţi de al doilea server este modelată de tranziţiile netemporizate t21 şi t22, având asignate probabilităţile p_{21}, respectiv p_{22}. Semnificaţia fizică a poziţiilor este evidentă.

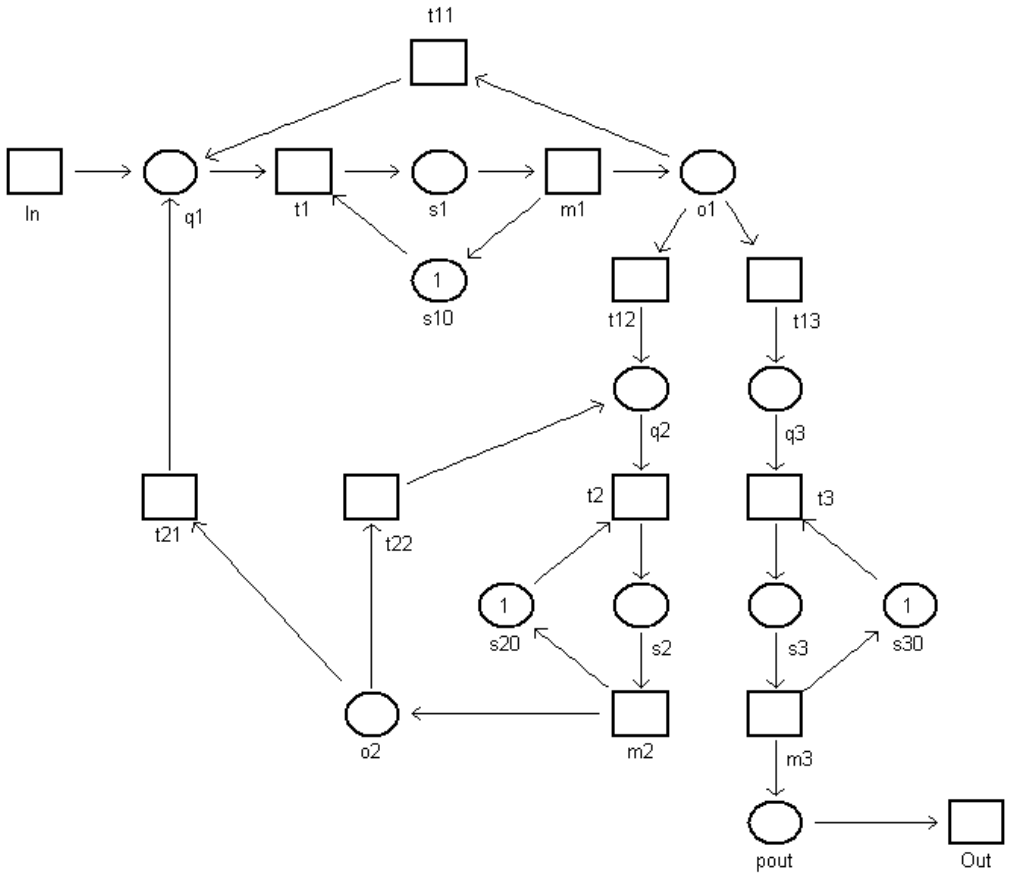


Fig. AP8.4.2. *Modelul de tip reţea Petri stohastică generalizată al sistemului de aşteptare din fig. AP8.4.1.*

2. Indicatorii globali referitori la pozițiile modelului din fig. AP8.4.2 determinați în urma simulării în mediul ***Petri Net Toolbox*** pe durata de 10.000 [s] sunt prezentate în fig. AP8.4.3.

(i). Rata de sosire a clienților la nodul *i* este dată de indicatorul ***Arrival Rate*** corespunzător poziției q*i* ce modelează firul de așteptare, $i = 1,2,3$, obținându-se 1,96 [s^{-1}] pentru nodul 1, 1,95 [s^{-1}] pentru nodul 2 și 0,99 [s^{-1}] pentru nodul 3.

(ii). Gradul de utilizare a serverului *i* este dat de indicatorul ***Queue Length*** corespunzător poziției s*i* ce modelează activitatea serverului, $i = 1,2,3$, obținându-se valorile 0,49 pentru nodul 1, 0,49 pentru nodul 2 și 0,48 pentru nodul 3.

(iii). Numărul mediu de clienți din firul de așteptare ce precede serverul *i* și durata medie de așteptare sunt date de indicatorii ***Queue Length*** și, respectiv, ***Waiting Time*** corespunzători poziției q*i*, $i = 1,2,3$.

Petri Net Toolbox - Web Browser

Global Statistics: Places

Model: ap8_4.xml

Events: 157236

Time: 10000.0711

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
q1	19633	1.9633	0.50935	19632	1.9632	0.50937	0.24327	0.47759
s1	19632	1.9632	0.50938	19632	1.9632	0.50937	0.25041	0.49161
o1	19632	1.9632	0.50938	19631	1.9632	0.50937	0	0
s10	19632	1.9632	0.50938	19632	1.9632	0.50937	0.25896	0.50839
q2	19555	1.9555	0.51138	19555	1.9561	0.51123	0.2397	0.46873
s2	19555	1.9555	0.51138	19555	1.956	0.51124	0.25226	0.4933
s20	19555	1.9555	0.51138	19555	1.9555	0.51138	0.25912	0.5067
o2	19555	1.9555	0.51138	19555	1.956	0.51125	0	0
q3	9919	0.99189	1.0082	9919	0.9919	1.0082	0.46147	0.45773
s3	9919	0.99189	1.0082	9918	0.9918	1.0083	0.49115	0.48712
s30	9918	0.99179	1.0083	9919	0.9919	1.0082	0.51707	0.51288
pout	9918	0.99179	1.0083	9918	0.9918	1.0083	0	0

Ok Save

Fig. AP8.4.3. *Indicatorii globali pentru pozițiile rețelei din fig. AP8.4.2.*

Observație: Performanțele rețelei de așteptare pot fi evaluate considerând fiecare nod separat ca fiind un sistem de așteptare de tip M/M/1 (Cassandras, 1993).

(i). Studiul analitic al dinamicii de regim permanent al acestei rețele de așteptare se bazează pe calcularea ratelor de sosire a clienților, λ_i, $i = \overline{1,3}$, la fiecare nod al rețelei în regim permanent. Scriind ecuațiile de bilanț al fluxului (eng. *flow balance equations*)

$$\begin{cases} \lambda_1 = r_1 + p_{11}\lambda_1 + p_{21}\lambda_2 \\ \lambda_2 = p_{12}\lambda_1 + p_{22}\lambda_2 \\ \lambda_3 = p_{13}\lambda_1 \end{cases}$$

rezultă

$$\lambda_1 = \frac{1}{p_{13}} r_1 = 2\,[\mathrm{s}^{-1}], \quad \lambda_2 = \frac{p_{12}}{p_{21}p_{13}} r_1 = 2\,[\mathrm{s}^{-1}], \quad \lambda_3 = r_1 = 1\,[\mathrm{s}^{-1}].$$

(ii). Gradele de utilizare a celor trei servere sunt $\rho_i = \lambda_i / \mu_i = 0.5 < 1$, $i = \overline{1,3}$, având valori subunitare, satisfac condiția de stabilitate şi confirmă faptul că rețeaua de aşteptare ajunge într-adevăr în regim permanent.

(iii). Numărul mediu de clienți la nodul *i* al rețelei fiind dat de $\mathrm{M}[X_i] = \dfrac{\rho_i}{1-\rho_i}$, $i = \overline{1,3}$, se obține $\mathrm{M}[X_1] = \mathrm{M}[X_2] = \mathrm{M}[X_3] = 1$.

Timpul mediu petrecut de un client la nodul *i* al rețelei se calculează aplicând legea lui Little:

$$\mathrm{M}[X_i] = \lambda_i \,\mathrm{M}[S_i] \quad \Rightarrow \quad \mathrm{M}[S_i] = \frac{1}{\mu_i(1-\rho_i)}, \quad i = \overline{1,3},$$

şi permite determinarea timpului mediu de aşteptare al unui client la acel nod

$$\mathrm{M}[W_i] = \mathrm{M}[S_i] - \mu_i, \quad i = \overline{1,3}.$$

Pentru valorile numerice precizate rezultă

$$\mathrm{M}[S_1] = \mathrm{M}[S_2] = 0.5\,[\mathrm{s}], \quad \mathrm{M}[S_3] = 1\,[\mathrm{s}],$$
$$\mathrm{M}[W_1] = \mathrm{M}[W_2] = 0.25\,[\mathrm{s}], \quad \mathrm{M}[W_3] = 0.5\,[\mathrm{s}].$$

Capitolul 9

Modele de tip max-plus

Breviar teoretic

BT9.1. Proprietăți fundamentale ale algebrei (max, +)

Considerăm mulțimea $\mathbb{R} \cup \{-\infty\}$, echipată cu o operație aditivă, notată $\oplus$, şi o operație multiplicativă, notată $\otimes$, definite după cum urmează:

$$a \oplus b = \max\{a, b\} \tag{BT9.1.1}$$

$$a \otimes b = a + b \tag{BT9.1.2}$$

unde operațiile binare 'max' şi '+' au semnificația standard cunoscută de către cititor.

Operația aditivă $\oplus$ posedă următoarele proprietăți:

1) Legea internă:

$$\forall a, b \in \mathbb{R} \cup \{-\infty\} : a \oplus b \in \mathbb{R} \cup \{-\infty\} \tag{BT9.1.3}$$

2) Comutativitatea:

$$\forall a, b \in \mathbb{R} \cup \{-\infty\} : a \oplus b = b \oplus a \tag{BT9.1.4}$$

3) Asociativitatea:

$$\forall a, b, c \in \mathbb{R} \cup \{-\infty\} : (a \oplus b) \oplus c = a \oplus (b \oplus c) \tag{BT9.1.5}$$

4) Existența elementului nul (neutru în raport cu $\oplus$):

$$\exists \varepsilon = -\infty \in \mathbb{R} \cup \{-\infty) : \forall a \in \mathbb{R} \cup \{-\infty\} : a \oplus \varepsilon = \varepsilon \oplus a = a \tag{BT9.1.6}$$

5) Idempotența:

$$\forall a \in \mathbb{R} \cup \{-\infty\} : a \oplus a = a \tag{BT9.1.7}$$

Operația multiplicativă $\otimes$ posedă următoarele proprietăți:

1) Legea internă:

$$\forall a, b \in \mathbb{R} \cup \{-\infty\} : a \otimes b \in \mathbb{R} \cup \{-\infty\}. \tag{BT9.1.8}$$

2) Comutativitatea:

$$\forall a, b \in \mathbb{R} \cup \{-\infty\} : a \otimes b = b \otimes a. \tag{BT9.1.9}$$

3) Asociativitatea:

$$\forall a, b, c \in \mathbb{R} \cup \{-\infty\} : (a \otimes b) \otimes c = a \otimes (b \otimes c). \tag{BT9.1.10}$$

4) Existența elementului unitate (neutru în raport cu $\otimes$):

$$\exists e = 0 \in \mathbb{R} \cup \{-\infty\} : \forall a \in \mathbb{R} \cup \{-\infty\} : a \otimes e = e \otimes a = a \,. \qquad \text{(BT9.1.11)}$$

5) Existența elementului invers:

$$\forall a \in \mathbb{R} \quad \exists a^{-1} = -a \in \mathbb{R} : a \otimes a^{-1} = a^{-1} \otimes a = e\,(=0) \quad . \qquad \text{(BT9.1.12)}$$

6) Elementul nul $\varepsilon = -\infty$ este absorbant în raport cu $\otimes$:

$$\forall a \in \mathbb{R} \quad a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon \qquad \text{(BT9.1.13)}$$

De asemenea, mai constatăm că operația multiplicativă $\otimes$ este distributivă în raport cu operația aditivă $\oplus$, adică:

$$\forall a,b,c \in \mathbb{R} \cup \{-\infty\} : (a \oplus b) \otimes c = a \otimes c \oplus b \otimes c \,. \qquad \text{(BT9.1.14)}$$

În baza proprietăților de mai sus, se poate afirma că mulțimea $\mathbb{R} \cup \{-\infty\}$ echipată cu operațiile $\oplus$ şi $\otimes$ are o structură algebrică de *semiinel idempotent comutativ* (Bacelli et al., 1992).

Operația de înmulțire $\otimes$ definită prin (BT9.1.2) permite introducerea operației *de ridicare la putere* a unui element din mulțimea $\mathbb{R} \cup \{-\infty\}$ prin relația

$$a^0 = e, \quad a^j = \underbrace{a \otimes \ldots \otimes a}_{j \text{ factori}} = \underbrace{a + \ldots + a}_{j \text{ termeni}} = ja, \quad j \in \mathbb{N},\ j \geq 1, \qquad \text{(BT9.1.15)}$$

ce poate fi apoi extinsă, în mod natural, la cazul

$$a^{1/j} = a/j, \quad j \in \mathbb{N},\ j \geq 1. \qquad \text{(BT9.1.16)}$$

BT9.2 Operaţii cu matrice şi vectori definiți pe algebra $(\mathbb{R} \cup \{-\infty\}, \max, +)$

Dacă $\boldsymbol{A}$, $\boldsymbol{B}$ sunt două matrice dreptunghiulare de aceeaşi dimensiune (m x n), ale căror elemente aparțin mulțimii $\mathbb{R} \cup \{-\infty\}$, definim *matricea sumă* în raport cu operația $\oplus$ prin:

$$\begin{aligned} &\boldsymbol{C} = \boldsymbol{A} \tilde{\oplus} \boldsymbol{B}, \\ &(\boldsymbol{C})_{ij} = (\boldsymbol{A})_{ij} \oplus (\boldsymbol{B})_{ij} = \max\left\{(\boldsymbol{A})_{ij}, (\boldsymbol{B})_{ij}\right\},\ i = 1,\ldots,m,\ j = 1,\ldots,n \end{aligned} \qquad \text{(BT9.2.1)}$$

Dacă $\boldsymbol{A}$, $\boldsymbol{B}$ sunt două matrice dreptunghiulare de dimensiuni (m x n) şi respectiv (n x p), ale căror elemente aparțin mulțimii $\mathbb{R} \cup \{-\infty\}$, definim *matricea produs* în raport cu operația $\tilde{\otimes}$ prin :

$$\begin{aligned} &\boldsymbol{C} = \boldsymbol{A} \tilde{\otimes} \boldsymbol{B}, \\ &(\boldsymbol{C})_{ij} = (\boldsymbol{A})_{i1} \otimes (\boldsymbol{B})_{1j} \oplus \ldots \oplus (\boldsymbol{A})_{in} \otimes (\boldsymbol{B})_{nj} = \\ &= \bigoplus_{k+1}^{n} \left((\boldsymbol{A})_{ik} \otimes (\boldsymbol{B})_{kj}\right) = \max\left\{(\boldsymbol{A})_{ik} + (\boldsymbol{B})_{kj}\right\}, \quad i = 1,\ldots,m,\ j = 1,\ldots,p. \end{aligned} \qquad \text{(BT9.2.2)}$$

Se observă cu uşurință că modul în care au fost definite matricele sumă şi produs, se supune întrutotul regulilor standard cunoscute de cititor, cu excepția faptului că operația aditivă $\oplus$ are semnificația de 'max', conform (BT9.1.1), iar operația multiplicativă $\otimes$ are semnificația de '+', conform (BT9.1.2). Operațiile cu vectori constituie un caz particular al celor discutate relativ la matrice.

Să notăm prin M_{nxn} mulțimea matricelor pătrate de ordin *n* cu elemente din $\mathbb{R} \cup \{-\infty\}$. Mulțimea M_{nxn} echipată cu operația $\tilde{\oplus}$ definită conform (BT9.2.1) posedă următoarele proprietăți:

1) Legea internă:

$$\forall \boldsymbol{A}, \boldsymbol{B} \in M_{nxn} : \boldsymbol{A} \tilde{\oplus} \boldsymbol{B} \in M_{nxn} . \qquad \text{(BT9.2.3)}$$

2) Comutativitate:

$$\forall \boldsymbol{A}, \boldsymbol{B} \in M_{nxn} : \boldsymbol{A} \tilde{\oplus} \boldsymbol{B} = \boldsymbol{B} \tilde{\oplus} \boldsymbol{A} . \qquad \text{(BT9.2.4)}$$

3) Asociativitate:

$$\forall \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in M_{nxn} : \left(\boldsymbol{A} \tilde{\oplus} \boldsymbol{B}\right) \tilde{\oplus} \boldsymbol{C} = \boldsymbol{A} \tilde{\oplus} \left(\boldsymbol{B} \tilde{\oplus} \boldsymbol{C}\right) . \qquad \text{(BT9.2.5)}$$

4) Existența matricei nule (neutre în raport cu $\oplus$):

$$\exists\, \boldsymbol{\varepsilon} \in M_{nxn,} \left(\boldsymbol{\varepsilon}\right)_{ij} = \boldsymbol{\varepsilon} = -\infty : \quad \forall \boldsymbol{A} \in M_{nxn} : \boldsymbol{A} \tilde{\oplus} \boldsymbol{\varepsilon} = \boldsymbol{\varepsilon} \tilde{\oplus} \boldsymbol{A} \qquad \text{(BT9.2.6)}$$

5) Idempotență:

$$\forall \ \ \boldsymbol{A} \in M_{nxn} : \boldsymbol{A} \tilde{\oplus} \boldsymbol{A} = \boldsymbol{A} . \qquad \text{(BT9.2.7)}$$

<u>*Observație:*</u> Proprietățile 1) - 5) enumerate anterior sunt valabile şi în cazul mai general al mulțimii matricelor dreptunghiulare.

Mulțimea M_{nxn} echipată cu operația $\tilde{\oplus}$ definită conform (BT9.2.2) posedă următoarele proprietăți:

1) Legea internă:

$$\forall \ \ \boldsymbol{A}, \boldsymbol{B} \in M_{nxn} : \boldsymbol{A} \tilde{\otimes} \boldsymbol{B} \in M_{nxn} . \qquad \text{(BT9.2.8)}$$

2) Asociativitate:

$$\forall \ \ \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in M_{nxn} : \left(\boldsymbol{A} \tilde{\otimes} \boldsymbol{B}\right) \tilde{\otimes} \boldsymbol{C} = \boldsymbol{A} \tilde{\otimes} \left(\boldsymbol{B} \tilde{\otimes} \boldsymbol{C}\right) . \qquad \text{(BT9.2.9)}$$

3) Existența matricei unitare (neutre în raport cu $\tilde{\otimes}$):

$$\exists \boldsymbol{E} \in M_{nxn}, \left(\boldsymbol{E}\right)_{ij} = e = 0, \left(E\right)_{ij} = \varepsilon = -\infty : \boldsymbol{A} \tilde{\otimes} \boldsymbol{E} = \boldsymbol{E} \tilde{\otimes} \boldsymbol{A} = \boldsymbol{A} . \qquad \text{(BT9.2.10)}$$

4) Matricea nulă $\boldsymbol{\varepsilon}$ este absorbantă în raport cu $\tilde{\otimes}$:

$$\forall \ \ \boldsymbol{A} \in M_{nxn} : \boldsymbol{A} \tilde{\otimes} \boldsymbol{\varepsilon} = \boldsymbol{\varepsilon} \tilde{\otimes} \boldsymbol{A} = \boldsymbol{\varepsilon} . \qquad \text{(BT9.2.11)}$$

De asemenea mai constatăm că operația multiplicativă $\tilde{\otimes}$ este distributivă în raport cu operația aditivă $\tilde{\oplus}$:

$$\forall \ \ \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in M_{nxn} : \left(\boldsymbol{A} \tilde{\oplus} \boldsymbol{B}\right) \tilde{\otimes} \boldsymbol{C} = \boldsymbol{A} \tilde{\otimes} \boldsymbol{C} \tilde{\oplus} \boldsymbol{B} \tilde{\otimes} \boldsymbol{C} . \qquad \text{(BT9.2.12)}$$

Pentru $\alpha \in \mathbb{R} \cup \{-\infty\}$ şi $\boldsymbol{A}$ o matrice dreptunghiulară (*m* x *n*) ale cărei elemente aparțin mulțimii $\mathbb{R} \cup \{-\infty\}$ definim operația (legea) externă:

$$\boldsymbol{B} = \alpha \hat{\otimes} \boldsymbol{A}, \ \left(\boldsymbol{B}\right)_{ij} = \alpha \otimes \left(\boldsymbol{A}\right)_{ij} = \alpha + \left(\boldsymbol{A}\right)_{ij}, \ i = 1, \ldots, m, \ j = 1, \ldots, n, \qquad \text{(BT9.2.13)}$$

care poartă denumirea de *înmulțire* a matricei $\boldsymbol{A}$ *cu scalarul* α.

Mulțimea M_{nxn} posedă următoarele proprietăți în raport cu operația externă de înmulțire cu scalari:

1) $$\forall \ \ \alpha \in \mathbb{R} \cup \{-\infty\}, \ \forall \ \ \boldsymbol{A} \in M_{nxn} : \alpha \hat{\otimes} \boldsymbol{A} \in M_{nxn} . \qquad \text{(BT9.2.14)}$$

2) $\forall\ \alpha,\beta\in\mathbb{R}\in\{-\infty\},\ \forall\ \boldsymbol{A}\in M_{nxn}:\alpha\hat{\otimes}\left(\beta\tilde{\otimes}\boldsymbol{A}\right)=\left(\alpha\otimes\beta\right)\hat{\otimes}\boldsymbol{A}$. (BT9.2.15)

3) $\forall\ \alpha,\beta\in\mathbb{R}\cup\{-\infty\},\ \forall\ \boldsymbol{A}\in M_{nxn}:\left(\alpha\oplus\beta\right)\hat{\otimes}\boldsymbol{A}=\alpha\hat{\otimes}\boldsymbol{A}\tilde{\oplus}\beta\hat{\otimes}\boldsymbol{A}$. (BT9.2.16)

4) $\forall\ \alpha\in\mathbb{R}\cup\{-\infty\},\ \forall\ \boldsymbol{A}\in M_{nxn}:\alpha\hat{\otimes}\left(\boldsymbol{A}\tilde{\oplus}\boldsymbol{B}\right)=\alpha\hat{\otimes}\boldsymbol{A}\tilde{\oplus}\alpha\hat{\otimes}\boldsymbol{B}$. (BT9.2.17)

5) $\forall\ \boldsymbol{A}\in M_{nxn}:e\hat{\otimes}\boldsymbol{A}=\boldsymbol{A}$. (BT9.2.18)

6) $\forall\ \boldsymbol{A}\in M_{nxn}:\varepsilon\hat{\otimes}\boldsymbol{A}=\boldsymbol{\varepsilon}$. (BT9.2.19)

Observaţie: Proprietăţile 1) – 6) enumerate anterior sunt valabile şi în cazul mai general al mulţimii matricelor dreptunghiulare.

În baza proprietăţilor de mai sus, se poate afirma că mulţimea M_{nxn} echipată cu operaţiile interne $\tilde{\oplus}$, $\tilde{\otimes}$ şi respectiv, cu operaţia externă $\hat{\otimes}$ în raport cu $\mathbb{R}\cup\{-\infty\}$, are o structură de *algebră idempotentă* (Bacelli et al., 1992).

Observaţie: Pentru a simplifica notaţiile, întrucât nu există pericol de confuzie începând din acest punct al expunerii, vom renunţa la simbolurile $\tilde{\oplus}$, $\tilde{\otimes}$, $\hat{\otimes}$ şi vom utiliza următoarea convenţie de scriere:

- $\oplus$ desemnează operaţia de adunare (BT9.1.1) din $\mathbb{R}\cup\{-\infty\}$ precum şi operaţia de adunare (BT2.1) din mulţimea matricelor.
- $\otimes$ desemnează operaţia de înmulţire (BT9.1.2) din $\mathbb{R}\cup\{-\infty\}$, operaţia de înmulţire (BT9.2.2) din mulţimea matricelor, precum şi operaţia de înmulţire a unei matrice cu un scalar din $\mathbb{R}\cup\{-\infty\}$ (BT9.2.13).

Sensul atribuit operaţiilor $\oplus$, $\otimes$ va reieşi cu uşurinţă din contextul în care acestea vor apărea.

Similar cazului clasic, al matricelor cu elemente din mulţimea numerelor reale, şi pentru o matrice pătratică $\boldsymbol{A}$ (de dimensiune $n\times n$) definită pe algebra max-plus se poate introduce *urma* acesteia, egală cu suma (în sensul algebrei max-plus) a elementelor de pe diagonala principală a matricei:

$$\text{trace}\,\boldsymbol{A}=\bigoplus_{i=1}^{n}\left(\boldsymbol{A}\right)_{ii}, \tag{BT9.2.20}$$

la care se va face apel ulterior în paragraful BT9.4.

BT9.3 Reprezentări de stare în algebra max-plus pentru reţele Petri de tip graf marcat, temporizate *P*

Fie $(\boldsymbol{N},\boldsymbol{M}_0)$ o reţea Petri de tip graf marcat, având *m* poziţii, ($p_j,\ j=1,\ldots,m$), n_s tranziţii sursă ($t_{i_s},\ i_s=1,\ldots,n_s$), n_r tranziţii receptor ($t_{i_r},\ i_r-1,\ldots,n_r$) şi *n* tranziţii care sunt conectate atât prin arce de intrare, cât şi prin arce de ieşire ($t_i,\ i=1,\ldots,n$).

Temporizarea de tip *P* alocă fiecărei poziţii p_j din graf durata $d_j\geq 0,\ j=1,\ldots,m$. Fiecare poziţie $p_j,\ j=1,\ldots,n$, a grafului marcat impune o disciplină FIFO (*First In First Out*)

pentru sosirea şi respectiv plecarea jetoanelor (plecare ce, evident, are loc după consumarea duratei de rezervare $d_j \geq 0$ aferentă poziției p_j).

Se consideră că toate jetoanele marcajului inițial sunt utilizabile la momentul de start (convențional luat $\tau_0 = 0$), adică duratele de timp aferente rezervării acestor jetoane au fost consumate înainte de $\tau_0 = 0$.

Pentru a desemna momentul când are loc cea de a *k* executare a unei tranziții din graf, se utilizează variabilele de mai jos:

$u_{i_s}(k)$ = precizează momentul de timp când tranziția sursă t_{i_s}, $i_s = 1,\ldots,n_s$, se execută pentru cea de-a *k* dată;

$y_{i_r}(k)$ = precizează momentul de timp când tranziția receptor t_{i_r}, $i_r = 1,\ldots,n_r$, se execută pentru cea de a *k* dată;

$x_i(k)$ = precizează momentul de timp când tranziția t_i, $i = 1,\ldots,n$, conectată atât prin arce de intrare cât şi prin arce de ieşire, se execută pentru cea de a *k* dată.

Observație: Variabilele $x_i(k)$, $y_{i_r}(k)$ desemnează cel mai devreme moment când tranzițiile respective se pot executa în dinamica grafului marcat, adică orice tranziție validată este imediat executată (fără nici o întârziere între validare şi executare).

Se construiesc următorii trei vectori ai căror elemente sunt momente de timp, cu semnificațiile detaliate anterior:

$$\boldsymbol{u}(k) = \begin{bmatrix} u_1(k) \\ \cdots \\ u_{n_s}(k) \end{bmatrix},\ \boldsymbol{x}(k) = \begin{bmatrix} x_1(k) \\ \cdots \\ x_n(k) \end{bmatrix},\ \boldsymbol{y}(t) = \begin{bmatrix} y_1(k) \\ \cdots \\ y_{n_r}(k) \end{bmatrix}. \qquad \text{(BT9.3.1)}$$

unde $\dim(\boldsymbol{u}) = n_s,\ \dim(\boldsymbol{x}) = n,\ \dim(\boldsymbol{y}) = n_r$.

Din modul de construcție a acestor vectori, rezultă inegalitățile vectoriale :

$$\forall\ k \in N : \boldsymbol{u}(k) \leq \boldsymbol{u}(k+1),\ \boldsymbol{x}(k) \leq \boldsymbol{x}(k+1),\ \boldsymbol{y}(k) \leq \boldsymbol{y}(k+1). \qquad \text{(BT9.3.2)}$$

Apelând la o scriere vectorial-matriceală şi utilizând operațiile $\oplus$ (BT9.2.1), $\otimes$ (BT9.2.2), între cei trei vectori (BT9.3.1), conținând momente de timp, pot fi evidențiate conexiuni de forma:

$$\begin{aligned} \boldsymbol{x}(k) &= \boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1) \oplus \ldots \oplus \boldsymbol{A}_M \otimes \boldsymbol{x}(k-M) \oplus \\ &\oplus \boldsymbol{B}_0 \otimes \boldsymbol{u}(k) \oplus \boldsymbol{B}_1 \otimes \boldsymbol{u}(k-1) \oplus \ldots \oplus \boldsymbol{B}_{M_s} \boldsymbol{u}(k-M_s) = \qquad \text{(BT9.3.3)} \\ &= \bigoplus_{l=0}^{M} \left(\boldsymbol{A}_l \otimes \boldsymbol{x}(k-l) \right) \oplus \bigoplus_{l_s=0}^{M_s} \left(\boldsymbol{B}_{l_s} \otimes \boldsymbol{u}(k-l_s) \right), \end{aligned}$$

$$\begin{aligned} \boldsymbol{y}(k) &= \boldsymbol{C}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{C}_1 \otimes \boldsymbol{x}(k-1) \oplus \ldots \oplus \boldsymbol{C}_{M_r} \otimes \boldsymbol{x}(k-M_r) \oplus \\ &\oplus \boldsymbol{D}_0 \otimes \boldsymbol{u}(k) \oplus \boldsymbol{D}_1 \otimes \boldsymbol{u}(k-1) \oplus \ldots \oplus \boldsymbol{D}_{M_{rs}} \otimes \boldsymbol{u}(k-M_{rs}) = \qquad \text{(BT9.3.4)} \\ &= \bigoplus_{l_r=0}^{M_r} \left(\boldsymbol{C}_{l_r} \otimes \boldsymbol{x}(k-l_r) \right) \oplus \bigoplus_{l_{rs}=0}^{M_{rs}} \left(\boldsymbol{D}_{l_{rs}} \otimes \boldsymbol{u}(k-l_{rs}) \right). \end{aligned}$$

unde:

M – notează numărul maxim de jetoane din marcajul iniţial ce există într-o poziţie precedată de o tranziţie din grupul (t_i) şi succedată tot de tranziţie din grupul (t_i);

M_s – notează numărul maxim de jetoane din marcajul iniţial ce există într-o poziţie precedată de o tranziţie din grupul (t_{i_s}) şi succedată de o tranziţie din grupul (t_i);

M_r – notează numărul maxim de jetoane din marcajul iniţial ce există într-o poziţie succedată de o tranziţie din grupul (t_{i_r}) şi precedată de o tranziţie din grupul (t_i);

M_{rs} – notează numărul maxim de jetoane din marcajul iniţial ce există într-o poziţie precedată de o tranziţie din grupul (t_{i_s}) şi succedată de o tranziţie din grupul (t_{i_r}).

Matricele implicate în descrierea (BT9.3.3), (BT9.3.4) au dimensiuni adecvate pentru a permite efectuarea calculelor.

Elementele acestor matrice sunt fie valori nenegative cu semnificaţia de durate de timp (selectate, în baza topologiei reţelei, din setul $dj \geq 0,\ j = 1,\ldots,m$, corespunzător temporizării *P*), fie elementul nul $\varepsilon = -\infty$ (BT9.1.6) (care marchează, după caz, neimplicarea în bilanţul temporal, a unora dintre momentele de timp conţinute în vectorii consideraţi).

În ecuaţiile (BT9.3.3) şi (BT9.3.4), valorile componentelor vectorilor $\boldsymbol{u}(1)$, $\boldsymbol{u}(2)$, $\boldsymbol{u}(3)$ etc. sunt cunoscute fiind date de momentele de timp când tranziţiile sursă (aparţinând grupului (t_{i_s})) se execută pentru prima, a doua, a treia dată, etc.

Pe seama celor două ecuaţii (BT9.3.3) şi (BT9.3.4) trebuie determinate valorile componentelor vectorilor $\boldsymbol{x}(1)$, $\boldsymbol{x}(2)$, $\boldsymbol{x}(3)$, etc. şi, respectiv $\boldsymbol{y}(1)$, $\boldsymbol{y}(2)$, $\boldsymbol{y}(3)$, etc.

Ecuaţiile (BT9.3.3) şi (BT9.3.4) definesc o *reprezentare de stare max-plus*, sau o *reprezentare intrare–stare–ieşire max-plus* asociată unui graf marcat cu temporizare deterministă de tip *P*.

Această denumire provine, prin analogie cu descrierile liniare utilizate în automatica clasică, analogie în baza căreia vectorii $\boldsymbol{u}(k)$, $\boldsymbol{x}(k)$, $\boldsymbol{y}(k)$ definiţi prin (BT9.3.1) sunt referiţi drept *vector de intrare*, *de stare* şi, respectiv, *de ieşire*. Atragem atenţia asupra faptului că analogia invocată are în vedere doar formalismul teoretic, dar nu şi semnificaţia fizică (deoarece în automatica clasică, componentele vectorilor $\boldsymbol{u}(k)$, $\boldsymbol{x}(k)$ şi $\boldsymbol{y}(k)$ reprezintă valori ale unor semnale, iar *k* desemnează variabila temporală de tip discret).

Prezenţa în ecuaţia (BT9.3.3) a vectorilor $\boldsymbol{x}(k-l),\ l = 1,\ldots,M$, şi $\boldsymbol{u}(k-l_s),\ l_s = 0,\ldots,M_s$, se explică prin faptul că momentul când o tranziţie din grupul $(t_i),\ i = 1,\ldots,n$, se execută pentru a *k*-a dată este legat de momentele de timp când celelalte tranziţii din grupul $(t_i),\ i = 1,\ldots,n$, şi/sau tranziţiile din grupul $(t_{i_s}),\ i_s = 1,\ldots,n_s$, se execută pentru a $k-l$, respectiv $k-l_s$ dată. Această legătură este impusă de marcajul iniţial al poziţiilor ce preced tranziţia din grupul $(t_i),\ i = 1,\ldots,n$, pentru care se exprimă momentul de timp când are loc cea de a k executare.

În mod similar se explică prezenţa în ecuaţia (BT9.3.4) a vectorilor $\boldsymbol{x}(k-l_r),\ l_r = 1,\ldots,M_r$, $\boldsymbol{u}(k-l_{rs}),\ l_{rs} = 0,\ldots,M_{rs}$. În exprimarea momentului de timp când o tranziţie din grupul (t_{i_r}) se execută pentru a k dată se ţine cont de numărul de jetoane existente în marcajul iniţial al fiecărei poziţii ce precede respectiva tranziţie.

În reprezentarea matriceal-vectorială (BT9.3.3) şi (BT9.3.4), o parte din matricele $\boldsymbol{A}_0, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_{M-1}$; $\boldsymbol{B}_0, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_{M_s-1}$, $\boldsymbol{C}_0, \boldsymbol{C}_1, \ldots, \boldsymbol{C}_{M_r-1}$; $\boldsymbol{D}_0, \boldsymbol{D}_1, \ldots, \boldsymbol{D}_{M_{rs}-1}$ pot avea toate elementele nule, $\varepsilon = -\infty$, motiv pentru care produsele ce includ aceste matrice se elimină la scrierea concretă a ecuațiilor (BT9.3.3) şi (BT9.3.4). Având în vedere semnificația elementelor nule $\varepsilon = -\infty$, într-o astfel de matrice putem face următoarele precizări:

- Pentru $M \geq 1$ o matrice $\boldsymbol{A}_l$, $l = 1, \ldots, M-1$, are toate elementele nule, dacă şi numai dacă toate pozițiile grafului marcat, precedate şi succedate de tranziții din grupul (t_i) au marcajul inițial diferit de *l*.
- Pentru $M_s \geq 1$ o matrice $\boldsymbol{B}_{l_s}$, $l_s = 0, \ldots, M_s - 1$, are toate elementele nule, dacă şi numai dacă toate pozițiile grafului marcat, precedate de tranziții din grupul (t_{i_s}) şi succedate de tranziții din grupul (t_i) au marcajul inițial diferit de l_s.
- Pentru $M_r \geq 1$ o matrice $\boldsymbol{C}_{l_r}$, $l_r = 0, \ldots, M_r - 1$, are toate elementele nule, dacă şi numai dacă toate pozițiile grafului marcat precedate de tranziții din grupul (t_i) şi succedate de tranziții din grupul (t_{i_s}) au marcajul inițial diferit de l_r.
- Pentru $M_{rs} \geq 1$, o matrice $\boldsymbol{D}_{l_{rs}}$, $l_{rs} = 0, \ldots, M_{rs} - 1$, are toate elementele nule, dacă şi numai dacă toate pozițiile grafului marcat precedate de o tranziție din grupul (t_{i_s}) şi succedate de o tranziție din grupul (t_{i_r}) au marcajul inițial diferit de l_{rs}.

Observație: În ecuația de stare (BT9.3.3) matricea $\boldsymbol{A}_0$ are întotdeauna diagonala formată din elemente nule, $\varepsilon = -\infty$, deoarece elementul generic $(\boldsymbol{A}_0)_{ii}$, $i = 1, \ldots, n$ serveşte în a lega momentul de timp $x_i(k)$ de el însuşi.

BT9.4. Rezolvarea ecuației de stare

Prin rezolvarea ecuației de stare (BT9.3.3) se determină global, în manieră vectorială, momentele de timp când se execută fiecare din tranzițiile din grupul (t_i), $i = 1, \ldots, n$, pentru cea de a *k* dată, conform semnificației vectorului $\boldsymbol{x}(k)$ definit prin (BT9.3.1).

Pentru *k* vom considera valori naturale, *k* = 1,2,3,..., ceea ce semnifică prima, a doua, a treia, etc. executare a tuturor tranzițiilor din grupul (t_i).

Înainte de a trece la procedura de rezolvare efectivă a ecuației (BT9.3.3), trebuie să observăm că această ecuație are o formă implicită în $\boldsymbol{x}(k)$ ori de câte ori matricea $\boldsymbol{A}_0$ conține elemente nenule (diferite de $\varepsilon = -\infty$). Cu alte cuvinte, constatăm că $\boldsymbol{x}(k)$, ca vector, este exprimat în funcție de el însuşi, în maniera:

$$\boldsymbol{x}(k) = \boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{v}(k) \qquad \text{(BT9.4.1)}$$

unde $\boldsymbol{v}(k)$ este un vector ce notează suma tuturor termenilor din membrul drept al ecuației (BT9.3.2), mai puțin $\boldsymbol{A}_0 \otimes \boldsymbol{x}(k)$, adică :

$$\boldsymbol{v}(k) = \bigoplus_{l=1}^{M} \left(\boldsymbol{A}_l \otimes \boldsymbol{x}(k-l) \right) \oplus \bigoplus_{l_s=0}^{M_s} \left(\boldsymbol{B}_{l_s} \otimes \boldsymbol{u}(k-l_s) \right). \qquad \text{(BT9.4.2)}$$

Soluția $\boldsymbol{x}(k)$ a ecuației (BT.4.1) este de forma:

$$\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k), \tag{BT9.4.3}$$

unde:

$$\boldsymbol{A}_0^* = \boldsymbol{E} \oplus \boldsymbol{A}_0 \oplus \boldsymbol{A}_0^2 \oplus \ldots \oplus \boldsymbol{A}_0^{n-1} = \bigoplus_{j=0}^{n-1} (\boldsymbol{A}_0)^j, \tag{BT9.4.4}$$

în care operația de ridicare la o putere naturală este definită pe baza operației de înmulțire prin:

$$(\boldsymbol{A}_0)^0 = \boldsymbol{E},\ (\boldsymbol{A}_0)^j = \underbrace{\boldsymbol{A}_0 \otimes \ldots \otimes \boldsymbol{A}_0}_{j \geq 1 \text{ factori}} \tag{BT9.4.5}$$

Observație: Forma soluției (BT9.4.3) a ecuației (BT9 4.1) rezultă prin înlocuirea succesivă a lui $\boldsymbol{x}(k)$ din membrul drept al lui (BT9.4.1) cu expresia lui însuşi $\boldsymbol{x}(k)$:

$$\begin{aligned}
\boldsymbol{x}(k) &= \boldsymbol{A}_0 \otimes x(k) \oplus \boldsymbol{v}(k), \\
\boldsymbol{x}(k) &= \boldsymbol{A}_0 \otimes \left[\boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{v}(k)\right] \oplus \boldsymbol{v}(k) \\
&= \boldsymbol{A}_0^2 \otimes \boldsymbol{x}(k) \oplus (\boldsymbol{E} \oplus \boldsymbol{A}_0) \otimes \boldsymbol{v}(k), \\
&\vdots \\
\boldsymbol{x}(k) &= \boldsymbol{A}_0^n \otimes \boldsymbol{x}(k) \oplus \left(\boldsymbol{E} \oplus \boldsymbol{A}_0 \oplus \boldsymbol{A}_0^2 \oplus \ldots \oplus \boldsymbol{A}_0^{n-1}\right) \otimes \boldsymbol{v}(k).
\end{aligned} \tag{BT9.4.6}$$

Cum toate elementele de pe diagonala lui $\boldsymbol{A}_0$ sunt nule, rezultă că $\boldsymbol{A}_0$ este nilpotentă măcar de ordinul *n*, adică $\boldsymbol{A}_0^n = \boldsymbol{\varepsilon}$ (toate elementele nule) şi ultima egalitate din (BT9.4.6) va furniza chiar soluția (BT9.4.3).

Construcția lui $\boldsymbol{A}_0^*$ introdusă în (BT9.4.4) defineşte *operatorul $*$ al lui Kleene*, care, în general are forma:

$$\boldsymbol{A}_0^* = \bigoplus_{j=0}^{\infty} (\boldsymbol{A}_0)^j, \tag{BT9.4.7}$$

dar în (BT9.4.4) avem $(\boldsymbol{A}_0)^j = \boldsymbol{\varepsilon}$ pentru $j \geq n$.

Întorcându-ne la rezolvarea ecuației (BT9.3.2), putem afirma că soluția acesteia este de forma:

$$\boldsymbol{x}(k) = \boldsymbol{v}(k), \text{ pentru } \boldsymbol{A}_0 = \boldsymbol{\varepsilon}, \tag{BT9.4.8-a}$$

şi respectiv:

$$\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k), \text{ pentru } \boldsymbol{A}_0 \neq \boldsymbol{\varepsilon}, \tag{BT9.4.8-b}$$

cu vectorul $\boldsymbol{v}(k)$ definit conform (BT9.4.2).

Considerând că rețeaua începe să funcționeze la momentul inițial, desemnat convențional drept $\tau_0 = 0$, examinarea ecuației (BT9.4.2) conduce la următoarele constatări:

- Pentru $1 \leq k \leq M$, $\boldsymbol{v}(k)$ include vectorii $\boldsymbol{x}(0)$, $\boldsymbol{x}(-1), \ldots, \boldsymbol{x}(k-M)$ pentru care toate componentele sunt nule, $\varepsilon = -\infty$, deoarece contorizarea în secvență naturală, prin intermediul lui *k*, a executărilor de tranziții din grupul (t_i) începe de la momentul $\tau_0 = 0$. Evident, pentru $M = 0$ vectorii în cauză nu apar pentru nici un *k*.

- Pentru $k>1$, $\boldsymbol{v}(k)$ include vectorii $\boldsymbol{x}(k-1)$, $\boldsymbol{x}(k-2),\ldots,\boldsymbol{x}(1)$ care au fost determinați la pașii anteriori şi au componente mai mari sau egale cu zero.
- Pentru $1\le k\le M_s$, $\boldsymbol{v}(k)$ include vectorii $\boldsymbol{u}(0)$, $\boldsymbol{u}(-1),\ldots,\boldsymbol{u}(k-M_s)$ pentru care toate componentele sunt nule, $\varepsilon=-\infty$, deoarece contorizarea în secvență naturală, prin intermediul lui k, a executărilor de tranziții din grupul (t_{i_s}) începe de la momentul $\tau_0=0$. Evident, pentru $M_s=0$ vectorii în cauză nu apar pentru nici un k.
- Pentru $k\ge 1$, $\boldsymbol{v}(k)$ include vectorii cunoscuți $\boldsymbol{u}(k)\ge\boldsymbol{u}(k-1)\ge\ldots\ge\boldsymbol{u}(1)\ge 0$, ale căror componente au valori numerice concrete.

Toate aceste constatări au un caracter general, depinzând de valoarea curentă a indicelui k, dar nu conțin nici o informație referitoare la executările de tranziții din grupul (t_i) care au loc chiar la momentul de start $\tau_0=0$ datorită marcajului inițial.

Să notăm prin $q_i\ge 0$ ordinul de validare al tranziției din grupul (t_i), $i=1,\ldots,n$, corespunzător marcajului inițial. Aceasta înseamnă că pentru fiecare tranziție t_i având $q_i\ge 1$, va trebui să obținem:

$$x_i(k)=0,\ 1\le k\le q_i. \qquad \text{(BT9.4.9)}$$

Pe de altă parte, fiecare tranziție t_i având $q_i\ge 1$ este precedată de poziții ce conțin cel puțin $q_i\le\max\{M,M_s\}$ jetoane; aceasta înseamnă că pentru $1\le k\le q_i$, scrierea implicită vectorial-matriceală (BT9.3.3) nu poate descrie executarea instantanee a tranzițiilor aflate în această situație. Altfel spus, satisfacerea condiției (BT9.4.9) nu poate fi obținută numai din valoarea $v_i(k)$, $1\le k\le q_i$, dată de (BT9.4.2) conform formulei (BT9.4.8). Drept urmare, formula (BT9.4.8) trebuie modificată corespunzător pentru a include cele q_i executări ale tranziției t_i, $i=1,\ldots,n$, care au loc la momentul de start $\tau_0=0$. Astfel, într-o formulare vectorială ce ține cont de toate componentele vectorului $\boldsymbol{x}(k)$, vom înlocui ecuațiile (BT9.4.8-a) şi (BT9.4.8-b) prin:

$$\boldsymbol{x}(k)=\bar{\boldsymbol{x}}(k)\oplus\boldsymbol{v}(k),\ \text{pentru}\ \ \boldsymbol{A}_0=\boldsymbol{\varepsilon} \qquad \text{(BT9.4.10-a)}$$

şi, respectiv:

$$\boldsymbol{x}(k)=\boldsymbol{A}_0^{*}\otimes\left(\bar{\boldsymbol{x}}(k)\oplus\boldsymbol{v}(k)\right),\ \text{pentru}\ \ \boldsymbol{A}_0\ne\boldsymbol{\varepsilon}, \qquad \text{(BT9.4.10-b)}$$

cu vectorul $\boldsymbol{v}(k)$ definit conform (BT9.4.2) şi vectorul $\bar{\boldsymbol{x}}(k)$ definit prin componentele sale, în următoarea manieră:

$$\bar{x}_i(k)=\begin{cases}0, & 1\le k\le q_i,\\ \varepsilon, & k>q_i.\end{cases} \qquad \text{(BT9.4.11)}$$

Evident, dacă tranziția t_i nu este validată la momentul de start $\tau_0=0$, atunci $\bar{x}_i(k)=\varepsilon$ pentru toate valorile k=1,2,3,… .

<u>Observație:</u> Dacă notăm prin $q_{x\max}$ ordinul maxim de validare al tranzițiilor din grupul (t_i), $i=1,\ldots,n$, adică

$$q_{x\max}=\max_{i=1,\ldots,n}\{q_i\} \qquad \text{(BT9.4.12)}$$

atunci, pentru orice $k>q_{x\max}$ vom avea $\bar{\boldsymbol{x}}(k)=\boldsymbol{\varepsilon}$, adică toate componentele lui $\bar{\boldsymbol{x}}(k)$ vor fi nule, $\varepsilon=-\infty$. Cu alte cuvinte, pentru $k>q_{x\max}$, formula (BT9.4.10) se reduce la formula

(BT9.4.8), vectorul $\bar{\boldsymbol{x}}(k)$ intervenind numai în primele $q_{x\max}$ iterații.

În particular, dacă la momentul de start $\tau_0 = 0$, nici o tranziție din grupul (t_i) nu este validată, atunci vom avea $\bar{\boldsymbol{x}}(k) = \boldsymbol{\varepsilon}$ pentru orice k=1,2,3,…, sau, altfel spus, formula (BT9.4.10) este echivalentă cu formula (BT9.4.8). Atragem atenția că acest caz particular nu poate apărea atunci când nu există tranziții de tipul (t_{i_s}) (tranziții sursă), deoarece într-o atare situație cel puțin una din tranzițiile din grupul (t_i), $i = 1,\ldots,n$, trebuie să fie validată la momentul de start $\tau_0 = 0$ pentru a asigura dinamica grafului marcat.

Caz particular: *Comportarea periodică a grafurilor marcate tare conexe, temporizate P*

După cum s-a prezentat în paragraful BT6.6, grafurile marcate tare conexe au o comportare ciclică, durata unui ciclu de operare fiind precizată prin formula (BT6.6.3).

Modelul de stare al unui graf marcat tare conex este reprezentat de:

$$\begin{aligned}\boldsymbol{x}(k) &= \boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1) \oplus \ldots \oplus \boldsymbol{A}_M \otimes \boldsymbol{x}(k-M) = \\ &= \bigoplus_{l=0}^{M} \left(\boldsymbol{A}_l \otimes \boldsymbol{x}(k-l)\right), \quad \text{pentru } \boldsymbol{A}_0 \neq \boldsymbol{\varepsilon},\end{aligned} \tag{BT9.4.13}$$

sau

$$\begin{aligned}\boldsymbol{x}(k) &= \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1) \oplus \ldots \oplus \boldsymbol{A}_M \otimes \boldsymbol{x}(k-M) = \\ &= \bigoplus_{l=1}^{M} \left(\boldsymbol{A}_l \otimes \boldsymbol{x}(k-l)\right), \quad \text{pentru } \boldsymbol{A}_0 = \boldsymbol{\varepsilon}.\end{aligned} \tag{BT9.4.14}$$

Ecuația implicită (BT9.4.13) poate fi, de asemenea, scrisă explicit:

$$\boldsymbol{x}(k) = \tilde{\boldsymbol{A}}_1 \otimes \boldsymbol{x}(k-1) \oplus \ldots \oplus \tilde{\boldsymbol{A}}_M \otimes \boldsymbol{x}(k-M) = \bigoplus_{l=1}^{M} \left(\tilde{\boldsymbol{A}}_l \otimes \boldsymbol{x}(k-l)\right), \tag{BT9.4.15}$$

unde $\tilde{\boldsymbol{A}}_l = \boldsymbol{A}_0^* \otimes \boldsymbol{A}_l$, $l = \overline{1, M}$.

Printr-o schimbare adecvată a vectorului de stare, ecuația (BT9.4.14) sau (BT9.4.15) poate fi în continuare adusă la forma:

$$\hat{\boldsymbol{x}}(k) = \boldsymbol{A} \otimes \hat{\boldsymbol{x}}(k-1), \tag{BT9.4.16}$$

în care matricea $\boldsymbol{A}$ are dimensiunea N x N, iar noul vector de stare $\hat{\boldsymbol{x}}(k)$ (cu N elemente) conține (eventual) şi unele dintre componentele vectorilor $\boldsymbol{x}(k-1)$, …, $\boldsymbol{x}(k-M)$.

În cazul considerat aici matricea $\boldsymbol{A}$ are o singură valoare proprie λ (Bacelli et al, 1992)

$$\lambda = \bigoplus_{j=1}^{N} \left(\operatorname{trace}\left(\boldsymbol{A}^j\right)\right)^{1/j}, \tag{BT9.4.17}$$

şi, eventual, mai mulți vectori proprii $\boldsymbol{x}$, care satisfac

$$\boldsymbol{A} \otimes \boldsymbol{x} = \lambda \otimes \boldsymbol{x}. \tag{BT9.4.18}$$

Vectorii $\hat{\boldsymbol{x}}(k)$, $k > 1$, corespunzători grafului marcat tare conex modelat prin ecuația (BT9.4.16), sunt vectori proprii asociați valorii proprii λ (BT9.4.17), astfel încât este satisfăcută relația:

$$\hat{\boldsymbol{x}}(k) = \lambda \otimes \hat{\boldsymbol{x}}(k-1) \quad \Leftrightarrow \quad \hat{x}_i(k) = \lambda + \hat{x}_i(k-1),\ i = \overline{1, N}. \tag{BT9.4.19}$$

Această relație arată că λ reprezintă tocmai *durata unui ciclu de operare* a grafului marcat tare conex luat în discuție.

BT9.5. Rezolvarea ecuației de ieşire

Prin rezolvarea ecuației de ieşire (BT9.3.4) se determină global, în maniera vectorială, momentele de timp când se execută fiecare din tranzițiile din grupul (t_{i_r}), $i=1,\ldots,n_r$, pentru cea de a k dată, conform semnificației vectorului $\boldsymbol{y}(k)$ definit prin (BT9.3.1). Pentru k vom considera valori naturale, k=1,2,3,…, ceea ce semnifică prima, a doua, a treia etc. executare a tuturor tranzițiilor din grupul (t_i).

Considerând că rețeaua începe să funcționeze la momentul inițial, desemnat convențional drept $\tau_0=0$, examinarea membrului drept al ecuației (BT9.3.4) conduce la următoarele constatări:

- Pentru $1\le k\le M_r$, $\boldsymbol{y}(k)$ include vectorii $\boldsymbol{x}(0)$, $\boldsymbol{x}(-1),\ldots,\boldsymbol{x}(k-M_r)$ pentru care toate componentele sunt nule $\varepsilon=-\infty$, deoarece contorizarea în secvență naturală, prin intermediul lui k, a executărilor de tranziții din grupul (t_i) începe la momentul $\tau_0=0$. Evident, pentru $M_r=0$, vectorii în cauză nu apar pentru nici un k.
- Pentru $k\ge 1$, $\boldsymbol{y}(k)$ include vectorii $\boldsymbol{x}(k)$, $\boldsymbol{x}(k-1),\ldots,\boldsymbol{x}(1)$ care au fost determinați din rezolvarea ecuației de stare (BT9.3.3).
- Pentru $1\le k\le M_{rs}$, $\boldsymbol{y}(k)$ include vectorii $\boldsymbol{u}(0)$, $\boldsymbol{u}(-1),\ldots,\boldsymbol{u}(k-M_{rs})$ pentru care toate componentele sunt nule, $\varepsilon=-\infty$, deoarece contorizarea în secvență naturală, prin intermediul lui k, a executărilor de tranziții din grupul t_{i_s} începe la momentul $\tau_0=0$. Evident, pentru $M_{rs}=0$, vectorii în cauză nu apar pentru nici un k.
- Pentru $k\ge 1$, $\boldsymbol{y}(k)$ include vectorii cunoscuți $\boldsymbol{u}(k)\ge\boldsymbol{u}(k-1)\ge\ldots\ge\boldsymbol{u}(1)\ge 0$, ale căror componente au valori numerice concrete.

Toate aceste constatări au un caracter general, depinzând de valoarea cunoscută a indicelui k, dar nu conțin nici o informație referitoare la executările de tranziții din grupul (t_{i_r}) care au loc la momentul de start $\tau_0=0$ datorită marcajului inițial.

Să notăm prin $q_{i_r}\ge 0$ ordinul de validare al tranziției (t_{i_r}), $i=1,\ldots,n_r$, corespunzător marcajului inițial. Aceasta înseamnă că pentru fiecare tranziție t_{i_r} având $q_{i_r}\ge 1$, va trebui să obținem:

$$y_{i_r}(k)=0,\ 1\le k\le q_{i_r} \tag{BT9.5.1}$$

Raționând la fel în cazul tranzițiilor din grupul (t_i), $i=1,\ldots,n$, luat în discuție în secțiunea precedentă, ajungem la concluzia că rezolvarea ecuației de ieşire (BT9.3.4) necesită adăugarea, în membrul drept al acesteia, a unui vector suplimentar $\overline{\boldsymbol{y}}(k)$, având componentele:

$$\overline{y}_{i_r}(k)=\begin{cases}0,\ 1\le k\le q_{i_r},\\ \varepsilon,\ k>q_{i_r},\end{cases}\quad i_r=1,\ldots,n_r, \tag{BT9.5.2}$$

care este responsabil de înregistrarea executărilor de tranziții datorate în exclusivitate marcajului inițial. În baza acestei concluzii, rezultă că ecuația de ieşire (BT9.3.4) se rezolvă cu ajutorul formulei cu scrierea vectorială:

$$\boldsymbol{y}(k)=\overline{\boldsymbol{y}}(k)\oplus\bigoplus_{l_r=0}^{M_r}\left(\boldsymbol{C}_{l_r}\otimes\boldsymbol{x}(k-l_r)\right)\oplus\bigoplus_{l_{rs}=0}^{M_{rs}}\left(\boldsymbol{D}_{l_{rs}}\otimes\boldsymbol{u}(k-l_{rs})\right). \tag{BT9.5.3}$$

Observaţie: Dacă notăm prin $q_{y\max}$ ordinul maxim de validare al tranziţiilor din grupul (t_{i_r}), $i_r = 1,\ldots,n_r$, adică

$$q_{y\max} = \max_{i_r=1,\ldots,n_r} \{q_{i_r}\}, \tag{BT9.5.4}$$

atunci, pentru orice $k > q_{y\max}$ vom avea $\overline{\boldsymbol{y}}(k) = \boldsymbol{\varepsilon}$, adică toate componentele lui $\overline{\boldsymbol{y}}(k)$ vor fi nule, $\varepsilon = -\infty$. Cu alte cuvinte, pentru $k > q_{y\max}$, formula (BT9.5.3) se reduce chiar la ecuaţia de ieşire (BT9.3.4), vectorul $\overline{\boldsymbol{y}}(k)$ intervenind numai în primele $q_{y\max}$ iteraţii. În particular, dacă la momentul de start $\tau_0 = 0$, nici o tranziţie din grupul (t_{i_r}) nu este validată, atunci vom avea $\overline{\boldsymbol{y}}(k) = \varepsilon$ pentru orice k = 1,2,3,…, sau, altfel spus, formula de rezolvare (BT9.5.3) este identică cu ecuaţia de ieşire (BT9.3.4).

Construcţia modelului max-plus este ilustrată prin **AP9.1** – **AP9.4**, rezolvarea ecuaţiei de stare este ilustrată prin **AP9.1** – **AP9.4**, iar rezolvarea ecuaţiei de ieşire este ilustrată prin **AP9.1** şi **AP9.4**.

Aplicaţii

AP9.1.

Se consideră reţelele Petri de tip graf marcat cu temporizare *P* reprezentate în fig. AP9.1.1.(a, b, c, d), în care durata de timp asignată poziţiei p_i este notată d_i, $i = \overline{1,4}$. Pentru fiecare dintre acestea să se abordeze următoarea problematică:

1. Să se construiască reprezentarea de stare max-plus asociată.
2. Să se expliciteze expresiile variabilelor de stare şi de ieşire de aşa manieră încât să se evidenţieze modul lor de calcul iterativ.

Soluţie

A. Pentru reţeaua din fig. AP 9.1.1.(a) se obţin următoarele rezultate:

1. Forma implicită a reprezentării de stare max-plus asociate reţelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon \\ d_2 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(t),$$

$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

2. Pentru matricea $\boldsymbol{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon \\ d_2 & \varepsilon \end{bmatrix}$ se obţine $\boldsymbol{A}_0^* = \boldsymbol{E} \oplus \boldsymbol{A}_0 = \begin{bmatrix} 0 & \varepsilon \\ d_2 & 0 \end{bmatrix}$ prin aplicarea operatorului Kleene. Cu aceasta, putem obţine $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k)$, unde $\boldsymbol{v}(k) = \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k)$.

Astfel avem:

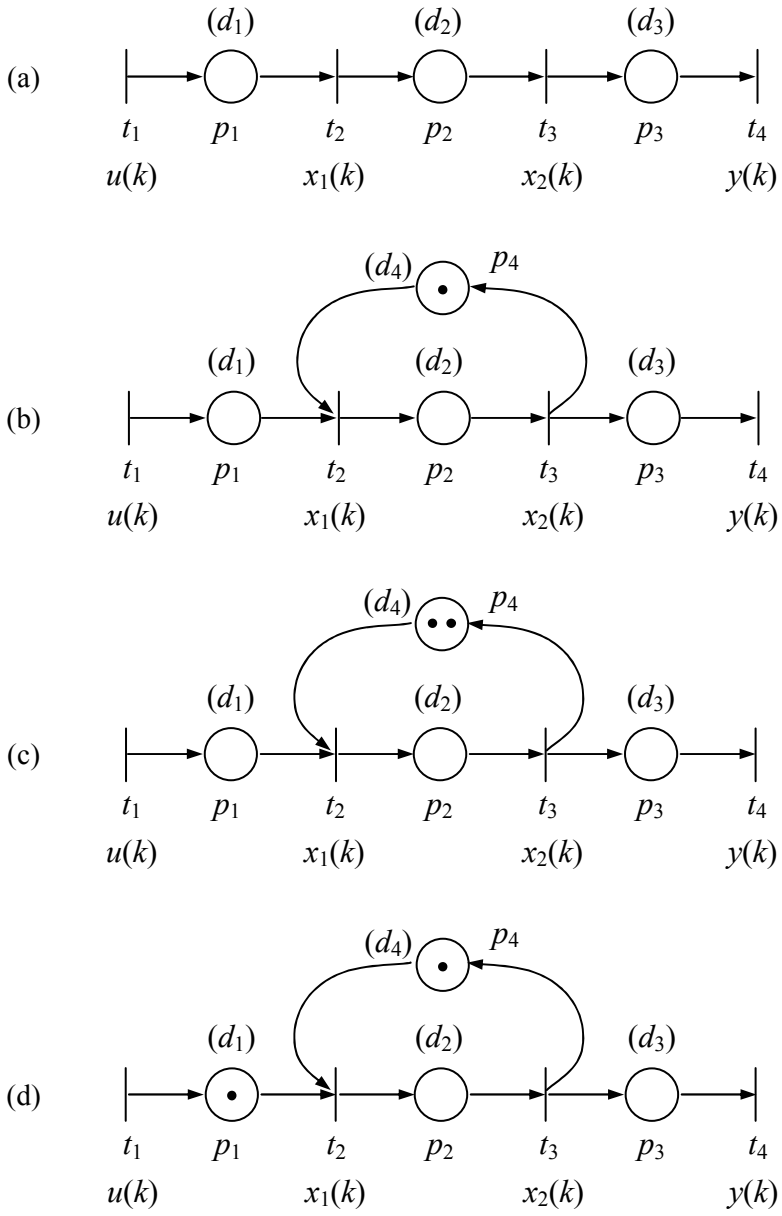


Fig. AP9.1.1. *Grafurile marcate temporizate P utilizate în AP9.1.*

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0 & \varepsilon \\ d_2 & 0 \end{bmatrix} \otimes \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k) = \begin{bmatrix} d_1 \\ d_1 \otimes d_2 \end{bmatrix} \otimes u(k),$$

$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} d_1 \\ d_1 \otimes d_2 \end{bmatrix} \otimes u(k) = \left(d_1 \otimes d_2 \otimes d_3 \right) \otimes u(k).$$

Ne aflăm în posesia tuturor informațiilor pentru a explicita formulele de calcul iterativ

$$\begin{aligned} x_1(k) &= d_1 \otimes u(k) = d_1 + u(k), \\ x_2(k) &= \left(d_1 \otimes d_2 \right) \otimes u(k) = \left(d_1 + d_2 \right) + u(k), \qquad k = 1, 2, \ldots . \\ y(k) &= \left(d_1 \otimes d_2 \otimes d_3 \right) \otimes u(k) = \left(d_1 + d_2 + d_3 \right) + u(k), \end{aligned}$$

B. Pentru reţeaua din fig. AP 9.1.1.(b) se obţin următoarele rezultate:
1. Forma implicită a reprezentării de stare max-plus asociate reţelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon \\ d_2 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(t),$$
$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

2. Cu aceleaşi matrice $\boldsymbol{A}_0$ şi $\boldsymbol{A}_0^*$ de la punctul A.2, din relaţia $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k)$ cu

$$\boldsymbol{v}(k) = \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k),$$

se obţine:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0 & \varepsilon \\ d_2 & 0 \end{bmatrix} \otimes \left\{ \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k) \right\},$$
$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

Pentru $k = 1$, ţinem cont de faptul că $x_1(0) = x_2(0) = \varepsilon$ şi, efectuând calculele, rezultă:

$$x_1(1) = d_1 \otimes u(1) = d_1 + u(1),$$
$$x_2(1) = (d_1 \otimes d_2) \otimes u(1) = (d_1 + d_2) + u(1),$$
$$y(1) = (d_3 \otimes (d_1 \otimes d_2)) \otimes u(1) = (d_1 + d_2 + d_3) + u(1).$$

Odată ce $x_1(1)$ şi $x_2(1)$ au fost determinaţi, pentru $k \geq 2$ se poate formula modul de calcul iterativ:

$$x_1(k) = d_4 \otimes x_2(k-1) \oplus d_1 \otimes u(k) = \max\{d_4 + x_2(k-1),\ d_1 + u(k)\},$$
$$x_2(k) = d_2 \otimes [d_4 \otimes x_2(k-1) \oplus d_1 \otimes u(k)] = d_2 + \max\{d_4 + x_2(k-1),\ d_1 + u(k)\},$$
$$y(k) = d_3 \otimes d_2 \otimes [d_4 \otimes x_2(k-1) \oplus d_1 \otimes u(k)] = (d_3 + d_2) + \max\{d_4 + x_2(k-1),\ d_1 + u(k)\},$$

pentru $k = 2, 3, \ldots$.

C. Pentru reţeaua din fig. AP 9.1.1.(c) se obţin următoarele rezultate:
1. Forma implicită a reprezentării de stare max-plus asociate reţelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon \\ d_2 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-2) \\ x_2(k-2) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(t),$$
$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

2. Cu aceleaşi matrice $\boldsymbol{A}_0$ şi $\boldsymbol{A}_0^*$ de la punctul A.2, putem scrie $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k)$ cu

$$\boldsymbol{v}(k) = \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-2) \\ x_2(k-2) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k).$$

Astfel avem:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0 & \varepsilon \\ d_2 & 0 \end{bmatrix} \otimes \left\{ \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-2) \\ x_2(k-2) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k) \right\},$$

$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

Pentru $k = 1$ şi $k = 2$ vom ţine cont de faptul că $x_1(-1) = x_2(-1) = \varepsilon$ şi respectiv, $x_1(0) = x_2(0) = \varepsilon$. Din efectuarea calculelor rezultă, pentru $k = 1$:

$$x_1(1) = d_1 \otimes u(1) = d_1 + u(1),$$
$$x_2(1) = (d_1 \otimes d_2) \otimes u(1) = (d_1 + d_2) + u(1),$$
$$y_3(1) = (d_3 \otimes (d_1 \otimes d_2)) \otimes u(1) = (d_1 + d_2 + d_3) + u(1),$$

şi respectiv, pentru $k = 2$:

$$x_1(2) = d_1 \otimes u(2) = d_1 + u(2),$$
$$x_2(2) = (d_1 \otimes d_2) d_2 \otimes u(1) = (d_1 + d_2) + u(2),$$
$$y_3(2) = (d_3 \otimes (d_1 \otimes d_2)) \otimes u(2) = (d_1 + d_2 + d_3) + u(2).$$

Odată ce $x_1(1)$, $x_2(1)$, $x_1(2)$ şi $x_2(2)$ au fost determinaţi, pentru $k \geq 3$ se poate formula modul de calcul iterativ:

$$x_1(k) = d_4 \otimes x_2(k-2) \oplus d_1 \otimes u(k) = \max\{d_4 + x_2(k-2),\ d_1 + u(k)\}, \qquad k = 3, 4, \ldots$$
$$x_2(k) = d_2 \otimes [d_4 \otimes x_2(k-2) \oplus d_1 \otimes u(k)] = d_2 + \max\{d_4 + x_2(k-2),\ d_1 + u(k)\},$$
$$y(k) = d_3 \otimes d_2 \otimes [d_4 \otimes x_2(k-2) \oplus d_1 \otimes u(k)] = (d_3 + d_2) + \max\{d_4 + x_2(k-2),\ d_1 + u(k)\}.$$

D. Pentru reţeaua din fig. AP 9.1.1.(d) se obţin următoarele rezultate:

1. Forma implicită a reprezentării de stare max-plus asociate reţelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon \\ d_2 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k-1),$$

$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

2. Cu aceleaşi matrice $\boldsymbol{A}_0$ şi $\boldsymbol{A}_0^*$ de la punctul A.2, putem scrie $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes [\overline{\boldsymbol{x}}(k) \oplus \boldsymbol{v}(k)]$ unde

$$\boldsymbol{v}(k) = \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ \varepsilon \end{bmatrix} \otimes u(k-1),$$

iar $\overline{\boldsymbol{x}}(k)$ este responsabil de executarea tranziţiilor asociate variabilelor de stare, ce sunt validate la momentul iniţial.

Astfel avem:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0 & \varepsilon \\ \varepsilon & 0 \end{bmatrix} \otimes \left\{ \begin{bmatrix} \overline{x}_1(k) \\ \overline{x}_2(k) \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & d_4 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \oplus \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \otimes u(k-1) \right\},$$

$$y(k) = \begin{bmatrix} \varepsilon & d_3 \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}.$$

Cum la momentul iniţial este validată numai tranziţia t_2 şi ordinul de validare este 1, înseamnă că pentru $\bar{\boldsymbol{x}}(k)$ putem scrie

$$\begin{bmatrix}\bar{x}_1(1)\\ \bar{x}_2(1)\end{bmatrix}=\begin{bmatrix}0\\ \varepsilon\end{bmatrix} \quad \text{şi} \quad \begin{bmatrix}\bar{x}_1(k)\\ \bar{x}_2(k)\end{bmatrix}=\begin{bmatrix}\varepsilon\\ \varepsilon\end{bmatrix},\ k\geq 2\,.$$

Totodată, pentru $k=1$ vom ţine cont de faptul că $x_1(0)=x_2(0)=\varepsilon$ şi $u_1(0)=\varepsilon$. Din efectuarea calculelor, rezultă, pentru $k=1$:

$$x_1(1)=0,$$
$$x_2(1)=d_2,$$
$$y_3(1)=d_3\otimes d_2=d_3+d_2.$$

Odată ce $x_1(1)$ şi $x_2(1)$ au fost determinaţi, pentru $k\geq 2$ se poate formula modul de calcul iterativ:

$$x_1(k)=d_4\otimes x_2(k-1)\oplus d_1\otimes u(k-1)=\max\left\{d_4+x_2(k-1),\ d_1+u(k-1)\right\},$$
$$x_2(k)=d_2\otimes\left[d_4\otimes x_2(k-1)\oplus d_1\otimes u(k-1)\right]=d_2+\max\left\{d_4+x_2(k-1),\ d_1+u(k-1)\right\},$$
$$y(k)=d_3\otimes d_2\otimes\left[d_4\otimes x_2(k-1)\oplus d_1\otimes u(k-1)\right]=\left(d_3+d_2\right)+\max\left\{d_4+x_2(k-1),\ d_1+u(k-1)\right\}.$$

pentru $k=2,3,\ldots$.

AP9.2.

Se consideră reţelele Petri din fig. AP9.2.1. Pentru fiecare dintre acestea să se abordeze următoarea problematică:

1. Să se construiască reprezentarea de stare max-plus asociată.
2. Să se expliciteze expresiile variabilelor de stare de aşa manieră încât să se evidenţieze modul lor de calcul iterativ.
3. Să se constate instalarea unui regim de funcţionare periodic şi să se precizeze perioada.

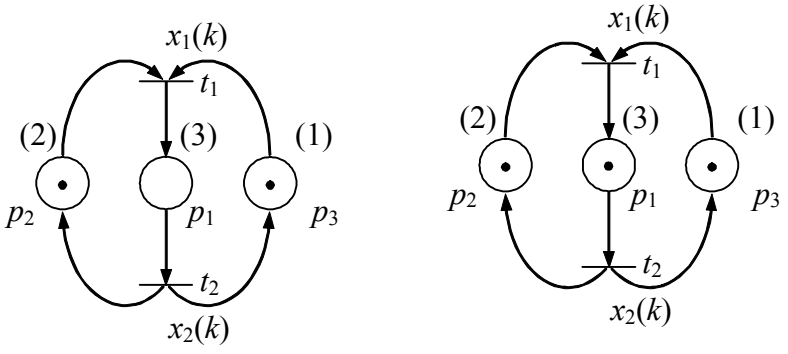


Fig. AP9.2.1. *Grafurile marcate temporizate P utilizate în AP9.2*

Soluţie

A. Pentru reţeaua din fig. AP 9.2.1.(a) se obţin următoarele rezultate:

1. Forma implicită a reprezentării de stare max-plus asociate reţelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon \\ 3 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & 2 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix}.$$

2. Aplicând operatorul Kleene matricei $\boldsymbol{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon \\ 3 & \varepsilon \end{bmatrix}$ se obține $\boldsymbol{A}_0^* = \boldsymbol{E} \oplus \boldsymbol{A}_0 = \begin{bmatrix} 0 & \varepsilon \\ 3 & 0 \end{bmatrix}$. Cu aceasta, putem scrie $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \left(\overline{\boldsymbol{x}}(k) \oplus \boldsymbol{v}(k) \right)$, unde $\boldsymbol{v}(k) = \begin{bmatrix} \varepsilon & 2 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix}$, iar $\overline{\boldsymbol{x}}(k)$ este responsabil de executarea tranzițiilor validate la momentul inițial.
Astfel avem:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0 & \varepsilon \\ 3 & 0 \end{bmatrix} \otimes \left\{ \begin{bmatrix} \overline{x}_1(k) \\ \overline{x}_2(k) \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & 2 \\ \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \right\}.$$

Cum la momentul inițial este validată numai tranziția t_1 şi ordinul de validare este 1, înseamnă că pentru $\overline{\boldsymbol{x}}(k)$ putem scrie:

$$\begin{bmatrix} \overline{x}_1(1) \\ \overline{x}_2(1) \end{bmatrix} = \begin{bmatrix} 0 \\ \varepsilon \end{bmatrix} \quad \text{şi} \quad \begin{bmatrix} \overline{x}_1(k) \\ \overline{x}_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, \; k \geq 2.$$

Totodată, pentru $k = 1$ vom ține cont de faptul că $x_1(0) = x_2(0) = \varepsilon$. Din efectuarea calculelor, rezultă, pentru $k = 1$:

$$x_1(1) = 0,$$
$$x_2(1) = 3.$$

Odată ce $x_1(1)$ şi $x_2(1)$ au fost determinați, pentru $k \geq 2$ se poate formula modul de calcul iterativ:

$$\begin{aligned} x_1(k) &= x_1(k-1) \oplus 2 \otimes x_2(k-1) = \max\left\{ x_1(k-1),\, 2 + x_2(k-1) \right\}, \\ x_2(k) &= 5 \otimes x_2(k-1) = 5 + x_2(k-1), \end{aligned} \qquad k = 2, 3, \ldots.$$

3. Dând valori lui *k*, observăm că executările tranzițiilor au loc la momentele de timp $x_1(k)$, $x_2(k)$:

k	1	2	3	4	…	k
$x_1(k)$	0	5	10	15	…	$5(k-1)$
$x_2(k)$	3	8	13	18	…	$5k-2$

Se constată instalarea unui regim periodic, cu perioada de 5 unități de timp.

B. Pentru rețeaua din fig. AP 9.2.1.(b) se obțin următoarele rezultate:
1. Reprezentarea de stare max-plus asociată rețelei este:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \varepsilon & 2 \\ 3 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix}.$$

2. Întrucât ecuația de stare rezultă direct în forma explicită (adică $\boldsymbol{x}(k)$ nu depinde de el însuşi ca în cazul rețelei din fig. AP9.2.1.(a)), se obține $\boldsymbol{x}(k) = \overline{\boldsymbol{x}}(k) \oplus \boldsymbol{v}(k)$, unde $\overline{\boldsymbol{x}}(k)$ este responsabil

de executarea tranziţiilor validate la momentul iniţial, iar $\boldsymbol{v}(k)=\begin{bmatrix}\varepsilon & 2\\ 3 & \varepsilon\end{bmatrix}\otimes\begin{bmatrix}x_1(k-1)\\ x_2(k-1)\end{bmatrix}$. Astfel avem:

$$\begin{bmatrix}x_1(k)\\ x_2(k)\end{bmatrix}=\begin{bmatrix}\overline{x}_1(k)\\ \overline{x}_2(k)\end{bmatrix}\oplus\begin{bmatrix}\varepsilon & 2\\ 3 & \varepsilon\end{bmatrix}\otimes\begin{bmatrix}x_1(k-1)\\ x_2(k-1)\end{bmatrix}.$$

Cum la momentul iniţial sunt validate ambele tranziţii t_1 şi t_2, iar ordinul de validare al fiecăreia dintre ele este 1, înseamnă că pentru $\overline{\boldsymbol{x}}(k)$ putem scrie:

$$\begin{bmatrix}\overline{x}_1(1)\\ \overline{x}_2(1)\end{bmatrix}=\begin{bmatrix}0\\ 0\end{bmatrix}\quad \text{şi}\quad \begin{bmatrix}\overline{x}_1(k)\\ \overline{x}_2(k)\end{bmatrix}=\begin{bmatrix}\varepsilon\\ \varepsilon\end{bmatrix},\ k\geq 2.$$

Totodată, pentru $k=1$ vom ţine cont de faptul că $x_1(0)=x_2(0)=\varepsilon$. Din efectuarea calculelor, rezultă, pentru k = 1:

$$x_1(1)=0,\qquad x_2(1)=0.$$

Odată ce $x_1(1)$ şi $x_2(1)$ au fost determinaţi, pentru $k\geq 2$ se poate formula modul de calcul iterativ:

$$\begin{aligned}x_1(k)&=2\otimes x_2(k-1)=2+x_2(k-1),\\ x_2(k)&=3\otimes x_1(k-1)=3+x_1(k-1),\end{aligned}\qquad k=2,3,\ldots.$$

3. Dând valori lui k, observăm că executările tranziţiilor au loc la momentele de timp

k	1	2	3	4	5	6	7	…	$2l$	$2l$+1
$x_1(k)$	0	2	5	7	10	12	15	…	5(l-1)+2	5l
$x_2(k)$	0	3	5	8	10	13	15	…	5(l-1)+3	5l

Se constată instalarea unui regim periodic, astfel că într-un interval de 5 [unităţi de timp] au loc câte două executări ale tranziţiilor t_1 şi t_2. Putem astfel considera că funcţionarea ciclică se realizează cu o perioadă $T=2{,}5$ [unităţi de timp] (în sensul că frecvenţa de executare a tranziţiilor reţelei este $f=0{,}4$ [unităţi de timp^{-1}]).

Observaţie: Aşa cum era de aşteptat, funcţionarea ciclică a celor două reţele evidenţiază aceleaşi valori pentru perioade care au fost determinate şi în contextul capitolului 6, anume **AP6.1**.

AP9.3.

Se consideră protocolul de comunicaţii a cărui funcţionare a fost studiată în **AP6.2** (păstrând aceleaşi valori pentru duratele de timp asignate tuturor poziţiilor).

1. Să se construiască reprezentarea de stare max-plus asociată.
2. Să se expliciteze expresiile variabilelor de stare şi de ieşire de aşa manieră încât să se evidenţieze modul lor de calcul iterativ.
3. Să se realizeze reprezentarea grafică a momentelor de timp la care începe transmisia unui mesaj de către emiţător (E) şi transmisia semnalului ACK de către receptor (R), utilizând mediul ***Petri Net Toolbox***.

Soluţie

1. Modelul utilizat pentru construirea reprezentării de stare max-plus a protocolului de comunicaţii studiat este prezentat în fig. AP9.3.1, în care sunt evidenţiate duratele de timp asignate poziţiilor reţelei, iar pentru desemnarea tranziţiilor t_i se foloseşte direct numele variabilelor de stare x_i, $i = 1,\ldots,6$.

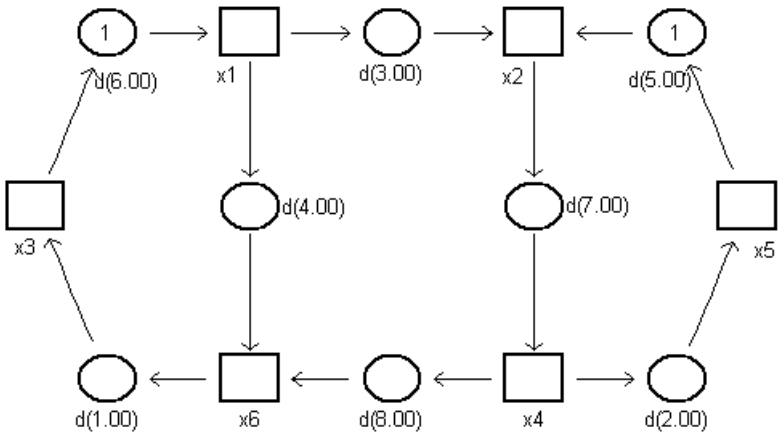


Fig. AP9.3.1. *Graful marcat cu temporizare P studiat în AP9.3.*

Considerând vectorul $\boldsymbol{x}(k) = \begin{bmatrix} x_1(k) & \cdots & x_6(k) \end{bmatrix}^{\mathrm{T}}$ având drept componente momentele de timp $x_i(k)$ la care se execută pentru cea de a k-a dată tranziţia t_i, $i = 1,\ldots,6$, modelul de stare în algebra max-plus, în formă implicită, este dat de:

$$\boldsymbol{x}(k) = \boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1),$$

unde:

$$\boldsymbol{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 \\ \varepsilon & 7 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon & \varepsilon \\ 4 & \varepsilon & \varepsilon & 8 & \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} \varepsilon & \varepsilon & 6 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 5 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}.$$

2. Vectorul $\boldsymbol{x}(k)$ se determină utilizând relaţia

$$\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \left(\bar{\boldsymbol{x}}(k) \oplus \boldsymbol{v}(k) \right), \quad k = 1,2,3,\ldots$$

unde $\boldsymbol{v}(k) = \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1)$, iar $\bar{\boldsymbol{x}}(k)$ este responsabil de executarea tranziţiilor validate la momentul iniţial. Singura tranziţie 1-validată la momentul iniţial $\tau_0 = 0$ este t_1, astfel încât vectorul $\bar{\boldsymbol{x}}(k)$ este dat de $\bar{\boldsymbol{x}}(1) = \begin{bmatrix} 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}^T$ şi $\bar{\boldsymbol{x}}(k) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}^T$, $k \geq 2$, în conformitate cu algoritmul prezentat în BT9.4.

Deoarece $\boldsymbol{x}(0)=\begin{bmatrix}\varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon\end{bmatrix}^T$, rezultă $\boldsymbol{x}(1)=\begin{bmatrix}0 & 3 & 19 & 10 & 12 & 18\end{bmatrix}^T$. Cunoscând $\boldsymbol{x}(1)$, în continuare se pot calcula valorile $\boldsymbol{x}(k)$, cu $k\geq 2$, după cum urmează. Se aplică operatorul Kleene matricei $\boldsymbol{A}_0$ şi se determină matricea $\boldsymbol{A}=\boldsymbol{A}_0^*\otimes\boldsymbol{A}_1$, obținându-se

$$\boldsymbol{A}_0^*=\begin{bmatrix}0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon\\ 3 & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon\\ 19 & 16 & 0 & 9 & \varepsilon & 1\\ 10 & 7 & \varepsilon & 0 & \varepsilon & \varepsilon\\ 12 & 9 & \varepsilon & 2 & 0 & \varepsilon\\ 18 & 15 & \varepsilon & 8 & \varepsilon & 0\end{bmatrix},\quad \boldsymbol{A}=\boldsymbol{A}_0^*\otimes\boldsymbol{A}_1=\begin{bmatrix}\varepsilon & \varepsilon & 6 & \varepsilon & \varepsilon & \varepsilon\\ \varepsilon & \varepsilon & 9 & \varepsilon & 5 & \varepsilon\\ \varepsilon & \varepsilon & 25 & \varepsilon & 21 & \varepsilon\\ \varepsilon & \varepsilon & 16 & \varepsilon & 12 & \varepsilon\\ \varepsilon & \varepsilon & 18 & \varepsilon & 14 & \varepsilon\\ \varepsilon & \varepsilon & 24 & \varepsilon & 20 & \varepsilon\end{bmatrix}.$$

Pentru valori $k\geq 2$, reprezentarea de stare poate fi scrisă în forma:

$$\boldsymbol{x}(k)=\boldsymbol{A}\otimes\boldsymbol{x}(k-1),\quad k\geq 2,$$

adică

$$\begin{cases}x_1(k)=6\otimes x_3(k-1)\\ x_2(k)=9\otimes x_3(k-1)\oplus 5\otimes x_5(k-1)\\ x_3(k)=25\otimes x_3(k-1)\oplus 21\otimes x_5(k-1)\\ x_4(k)=16\otimes x_3(k-1)\oplus 12\otimes x_5(k-1)\\ x_5(k)=18\otimes x_3(k-1)\oplus 14\otimes x_5(k-1)\\ x_6(k)=24\otimes x_3(k-1)\oplus 20\otimes x_5(k-1)\end{cases}\Leftrightarrow\begin{cases}x_1(k)=6+x_3(k-1)\\ x_2(k)=\max\{9+x_3(k-1),\ 5+x_5(k-1)\}\\ x_3(k)=\max\{25+x_3(k-1),\ 21+x_5(k-1)\}\\ x_4(k)=\max\{16+x_3(k-1),\ 12+x_5(k-1)\}\\ x_5(k)=\max\{18+x_3(k-1),\ 14+x_5(k-1)\}\\ x_6(k)=\max\{24+x_3(k-1),\ 20+x_5(k-1)\}\end{cases}$$

Prin inducție matematică se demonstrează că $x_3(k)>x_5(k)$, pentru $k=1,2,\ldots$, astfel încât, în final, rezultă:

$$\begin{aligned}&x_1(k)=6+x_3(k-1),\\ &x_2(k)=9+x_3(k-1),\\ &x_3(k)=25+x_3(k-1),\\ &x_4(k)=16+x_3(k-1),\\ &x_5(k)=18+x_3(k-1),\\ &x_6(k)=24+x_3(k-1),\end{aligned}\quad\text{pentru}\quad k=2,3,\ldots.$$

Pentru primele 5 iterații ale algoritmului se obțin următoarele valori numerice:

$$\boldsymbol{x}(1)=\begin{bmatrix}0\\3\\19\\10\\12\\18\end{bmatrix},\ \boldsymbol{x}(2)=\begin{bmatrix}25\\28\\44\\35\\37\\43\end{bmatrix},\ \boldsymbol{x}(3)=\begin{bmatrix}50\\53\\69\\60\\62\\68\end{bmatrix},\ \boldsymbol{x}(4)=\begin{bmatrix}75\\78\\94\\85\\87\\93\end{bmatrix},\ \boldsymbol{x}(5)=\begin{bmatrix}100\\103\\119\\110\\112\\118\end{bmatrix}.$$

Se observă că sistemul are o comportare periodică cu perioada de 25 [unități de timp]. Analitic, periodicitatea comportării sistemului se demonstrează pornind de la relația $x_3(k)=25+x_3(k-1)$, $k\geq 2$, de unde rezultă relația vectorială $\boldsymbol{x}(k)=25\otimes\boldsymbol{0}\oplus\boldsymbol{x}(k-1)$, cu

$\boldsymbol{0} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$ (reamintim că în algebra max-plus, 0 este elementul neutru față de operația multiplicativă $\otimes$).

O altă posibilitate de determinare a perioadei este prin calcularea valorii proprii a matricei $\boldsymbol{A}$ utilizând relația (BT9.4.17), pe care o lăsăm cititorului drept exercițiu.

3. Mediul ***Petri Net Toolbox*** oferă posibilitatea vizualizării descrierii de stare în algebra max-plus asociată unui graf marcat temporizat *P* (fig. AP9.3.2), precum şi a reprezentărilor grafice $x_1(k)$, $x_4(k)$, pentru $k = 1,2,...,5$ (fig. AP9.3.3).

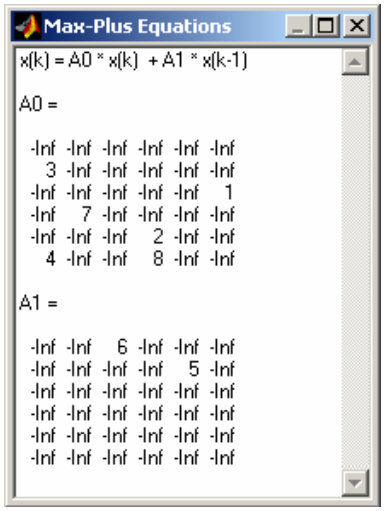

Max-Plus Equations
x(k) = A0 * x(k) + A1 * x(k-1)
A0 =
-Inf -Inf -Inf -Inf -Inf -Inf
3 -Inf -Inf -Inf -Inf -Inf
-Inf -Inf -Inf -Inf -Inf 1
-Inf 7 -Inf -Inf -Inf -Inf
-Inf -Inf -Inf 2 -Inf -Inf
4 -Inf -Inf 8 -Inf -Inf
A1 =
-Inf -Inf 6 -Inf -Inf -Inf
-Inf -Inf -Inf -Inf 5 -Inf
-Inf -Inf -Inf -Inf -Inf -Inf
-Inf -Inf -Inf -Inf -Inf -Inf
-Inf -Inf -Inf -Inf -Inf -Inf
-Inf -Inf -Inf -Inf -Inf -Inf

Fig. AP9.3.2. *Reprezentarea de stare max-plus asociată rețelei din fig. AP9.3.1.*

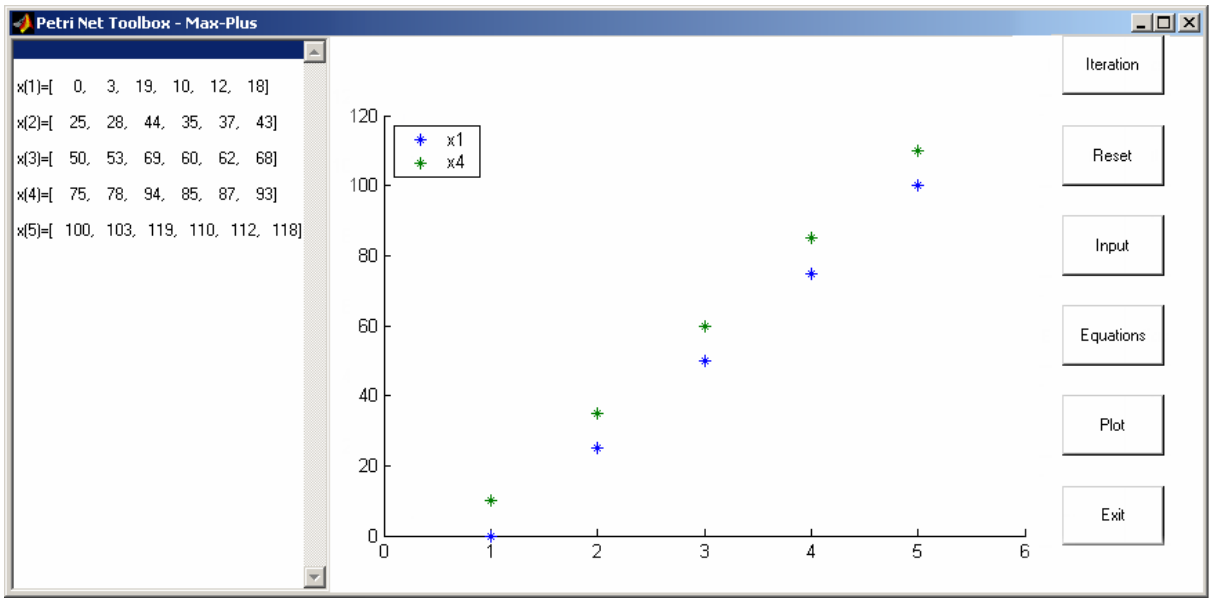


Fig. AP9.3.3. *Reprezentările grafice* $x_1(k)$, $x_4(k)$, *pentru* $k = \overline{1,5}$, *corespunzătoarei rețelei din fig. AP9.3.1 afişate în fereastra deschisă de* ***Petri Net Toolbox*** *pentru analiza max-plus.*

AP9.4.

Se consideră o celulă de fabricație flexibilă – adaptare după (Bacelli et al., 1992) – ce constă din trei maşini de prelucrare universale M_1, M_2 şi M_3. Se fabrică 3 tipuri de produse, P_1, P_2 şi P_3, care necesită următoarele secvențe de prelucrări (fig. AP9.4.1):

- pentru produsul P_1: prelucrare 1 pe M_2 (durata $d_{12} = 3$ min), prelucrare 2 pe M_3 (durata $d_{13} = 4$ min);
- pentru produsul P_2: prelucrare 1 pe M_1 (durata $d_{21} = 1$ min), prelucrare 2 pe M_2 (durata $d_{22} = 2$ min), prelucrare 3 pe M_3 (durata $d_{23} = 3$ min);
- pentru produsul P_3: prelucrare 1 pe M_1 (durata $d_{31} = 5$ min), prelucrare 2 pe M_2 (durata $d_{32} = 3$ min).

Duratele pentru transportul pieselor şi pentru eliberarea maşinilor sunt considerate neglijabile.
Se consideră următoarea strategie de planificare a servirii produselor (clienților) P_1, P_2, P_3 de către resursele M_1, M_2, M_3:

(S) M_1: P_2, P_3; M_2: P_1, P_2, P_3; M_3: P_1, P_2.

1. Să se construiască modelul tip graf marcat temporizat *P* al operațiilor din celula de fabricație.
2. Să se construiască reprezentarea de stare max-plus asociată.
3. Să se expliciteze expresiile variabilelor de stare şi de ieşire de aşa manieră încât să se evidențieze modul lor de calcul iterativ.
4. Considerând că celula de fabricație este alimentată cu piese de tipul P_1 la momentele 0, 3, 10, 15, 16, 25 [min], de tipul P2 la momentele 0, 4, 4, 10, 20, 21 [min] şi de tipul P_3 la momentele 4, 9, 13, 15, 22, 29 [min], să se determine momentele de timp la care sunt finalizate piesele de tipul P_1, P_2 şi, respectiv, P_3.
5. Utilizând mediul ***Petri Net Toolbox***, să se reprezinte grafic momentele de timp la care cele 6 piese de tipul P_1 sunt introduse în sistem, încep prelucrarea pe maşina M_2, încep prelucrarea pe maşina M_3 şi sunt degajate din celula de fabricație ca produse finite.

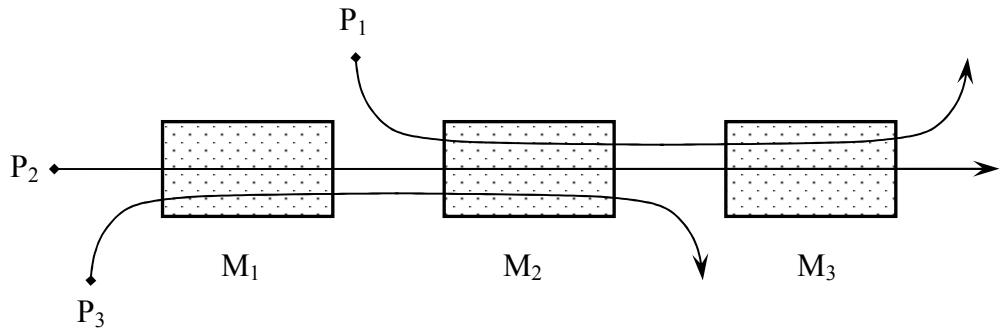


Fig. AP9.4.1. *Succesiunea de prelucrări în celula de fabricație din AP9.4.*

<u>*Soluție*</u>

1. În fig. AP9.4.2 este prezentat modelul de tip graf marcat cu temporizare *P* al celulei de fabricație. Pozițiilor le sunt asignate durate de timp în concordanță cu semnificațiile lor fizice având valorile precizate în figură. Marcajul inițial al rețelei plasează jetoane numai pe pozițiile ce modelează disponibilitatea maşinilor şi satisface condițiile impuse prin strategia

(S) de planificare a servirii produselor. Pentru desemnarea tranzițiilor se utilizează direct numele variabilelor de intrare u_{i_s}, $i_s = 1,2,3$, de stare x_i, $i = 1,\ldots,7$, şi de ieşire y_{i_r}, $i_r = 1,2,3$.

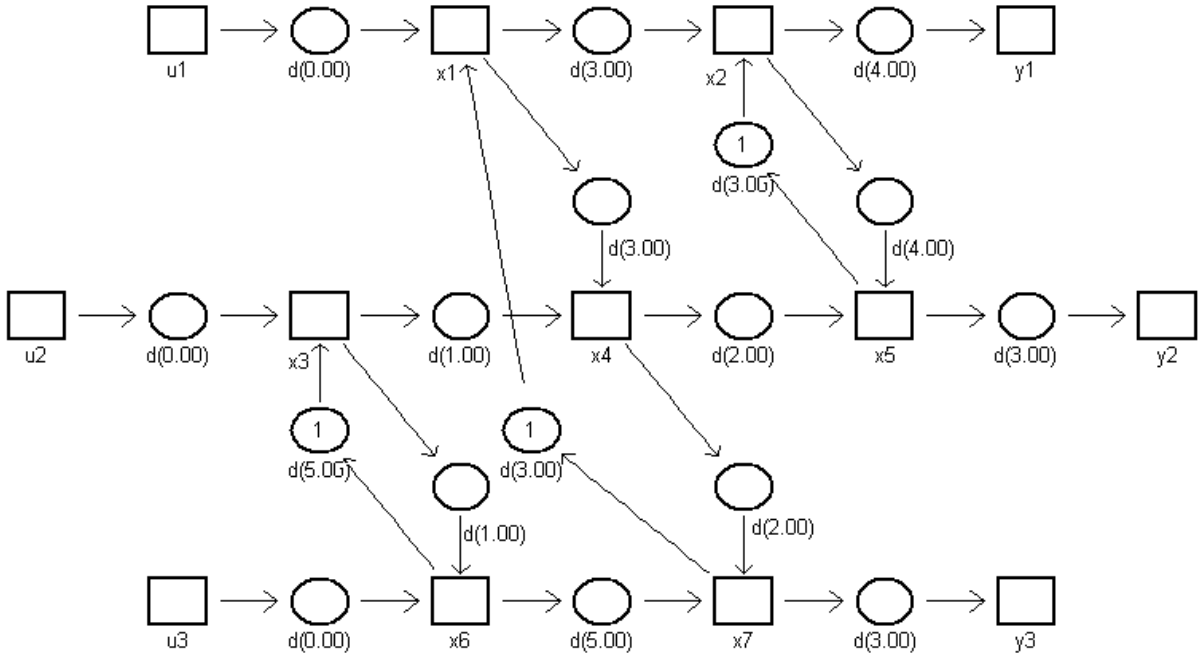


Fig. AP9.4.2. *Modelul de tip graf marcat cu temporizare P utilizat în AP9.4.*

2. Graful marcat din fig. AP9.4.2 are 3 tranziții sursă (u1, u2 şi u3), 3 tranziții receptor (y1, y2 şi y3), precum şi 7 tranziții conectate atât prin arce de intrare cât şi prin arce de ieşire (x1, …, x7). Considerăm vectorii $\boldsymbol{u}(k) = \begin{bmatrix} u_1(k) & u_2(k) & u_3(k) \end{bmatrix}^T$, $\boldsymbol{y}(k) = \begin{bmatrix} y_1(k) & y_2(k) & y_3(k) \end{bmatrix}^T$ şi $\boldsymbol{x}(k) = \begin{bmatrix} x_1(k) & \ldots & x_7(k) \end{bmatrix}^{\mathrm{T}}$ cu semnificațiile din paragraful BT9.3.

Reprezentarea de stare max-plus a acestui graf marcat este

$$\boldsymbol{x}(k) = \boldsymbol{A}_0 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0 \otimes \boldsymbol{u}(k),$$

$$\boldsymbol{y}(k) = \boldsymbol{C}_0 \otimes \boldsymbol{x}(k),$$

pentru care matricele implicate au următoarele valori:

$$\boldsymbol{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3 & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 4 & \varepsilon & 2 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon & 5 & \varepsilon \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 5 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{B}_0 = \begin{bmatrix} 0 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 0 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 0 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix},$$

$$\boldsymbol{C}_0 = \begin{bmatrix} \varepsilon & 4 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 \end{bmatrix}.$$

Este evident faptul că momentele de timp la care se execută tranzițiile sursă, cu ajutorul cărora se construiesc vectorii $\boldsymbol{u}(k)$, $k = 1, 2, \ldots$, sunt considerate cunoscute.

3. Rezolvarea ecuației de stare urmăreşte algoritmul prezentat în paragraful BT9.4. Deoarece nici una dintre tranzițiile rețelei nu este validată la momentul inițial $\tau_0 = 0$, vectorul de stare este dat de $\boldsymbol{x}(k) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(k)$, unde

$$\boldsymbol{A}_0^* = \begin{bmatrix} 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3 & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3 & \varepsilon & 1 & 0 & \varepsilon & \varepsilon & \varepsilon \\ 7 & 4 & 3 & 2 & 0 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & 0 & \varepsilon \\ 5 & \varepsilon & 6 & 2 & \varepsilon & 5 & 0 \end{bmatrix}.$$

şi $\boldsymbol{v}(k) = \boldsymbol{A}_1 \otimes \boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0 \otimes \boldsymbol{u}(k)$.

Pentru $k = 1$, ținem cont de faptul că $x_1(0) = \ldots = x_7(0) = \varepsilon$ şi, efectuând calculele, se obține $\boldsymbol{v}(1) = \boldsymbol{B}_0 \otimes \boldsymbol{u}(1) = \begin{bmatrix} u_1(1) & \varepsilon & u_2(1) & \varepsilon & \varepsilon & u_3(1) & \varepsilon \end{bmatrix}^T$. Vectorul $\boldsymbol{x}(1) = \boldsymbol{A}_0^* \otimes \boldsymbol{v}(1)$ rezultă de forma:

$$\boldsymbol{x}(1) = \begin{bmatrix} u_1(1) \\ 3 \otimes u_1(1) \\ u_2(1) \\ 3 \otimes u_1(1) \oplus 1 \otimes u_2(1) \\ 7 \otimes u_1(1) \oplus 3 \otimes u_2(1) \\ 1 \otimes u_2(1) \oplus 0 \otimes u_3(1) \\ 5 \otimes u_1(1) \oplus 6 \otimes u_2(1) \oplus 5 \otimes u_3(1) \end{bmatrix} \Leftrightarrow \begin{cases} x_1(1) = u_1(1) \\ x_2(1) = 3 + u_1(1) \\ x_3(1) = u_2(1) \\ x_4(1) = \max\{3 + u_1(1),\ 1 + u_2(1)\} \\ x_5(1) = \max\{7 + u_1(1),\ 3 + u_2(1)\} \\ x_6(1) = \max\{1 + u_2(1),\ u_3(1)\} \\ x_7(1) = \max\{5 + u_1(1),\ 6 + u_2(1),\ 5 + u_3(1)\} \end{cases}.$$

Pentru $k \geq 2$, considerând matricele

$$\boldsymbol{A} = \boldsymbol{A}_0^* \otimes \boldsymbol{A}_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3 & \varepsilon & 6 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 5 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 6 & 6 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 7 & 8 & 10 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 6 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 11 & 8 \end{bmatrix}, \quad \boldsymbol{B} = \boldsymbol{A}_0^* \otimes \boldsymbol{B}_0 = \begin{bmatrix} 0 & \varepsilon & \varepsilon \\ 3 & \varepsilon & \varepsilon \\ \varepsilon & 0 & \varepsilon \\ 3 & 1 & \varepsilon \\ 7 & 3 & \varepsilon \\ \varepsilon & 1 & 0 \\ 5 & 6 & 5 \end{bmatrix},$$

ecuația de stare ia forma $\boldsymbol{x}(k) = \boldsymbol{A} \otimes \boldsymbol{x}(k-1) \oplus \boldsymbol{B} \otimes \boldsymbol{u}(k)$, care poate fi direct utilizată pentru calcularea valorilor vectorilor $\boldsymbol{x}(k)$, $k \geq 2$, anume

$$\boldsymbol{x}(k)=\begin{bmatrix} 3\otimes x_7(k-1)\oplus u_1(k) \\ 3\otimes x_5(k-1)\oplus 6\otimes x_7(k-1)\oplus 3\otimes u_1(k) \\ 5\otimes x_6(k-1)\oplus u_2(k) \\ 6\otimes x_6(k-1)\oplus 6\otimes x_7(k-1)\oplus 3\otimes u_1(k)\oplus 1\otimes u_2(k) \\ 7\otimes x_5(k-1)\oplus 8\otimes x_6(k-1)\oplus 10\otimes x_7(k-1)\oplus 7\otimes u_1(k)\oplus 3\otimes u_2(k) \\ 6\otimes x_6(k-1)\oplus 1\otimes u_2(k)\oplus u_3(k) \\ 11\otimes x_6(k-1)\oplus 8\otimes x_7(k-1)\oplus 5\otimes u_1(k)\oplus 6\otimes u_2(k)\oplus 5\otimes u_3(k) \end{bmatrix} \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} x_1(k)=\max\{3+x_7(k-1),\, u_1(k)\} \\ x_2(k)=\max\{3+x_5(k-1),\, 6+x_7(k-1),\, 3+u_1(k)\} \\ x_3(k)=\max\{5+x_6(k-1),\, u_2(k)\} \\ x_4(k)=\max\{6+x_6(k-1),\, 6+x_7(k-1),\, 3+u_1(k),\, 1+u_2(k)\} \\ x_5(k)=\max\{7+x_5(k-1),\, 8+x_6(k-1),10+x_7(k-1),7+u_1(k),3+u_2(k)\} \\ x_6(k)=\max\{6+x_6(k-1),\, 1+u_2(k),\, u_3(k)\} \\ x_7(k)=\max\{11+x_6(k-1),\, 8+x_7(k-1),\, 5+u_1(k),\, 6+u_2(k),\, 5+u_3(k)\} \end{cases}.$$

Cunoscând $\boldsymbol{x}(k),\ k\geq 1$, ecuația de ieşire furnizează $\boldsymbol{y}(k)=\boldsymbol{C}_0\otimes\boldsymbol{x}(k)$, adică

$$\boldsymbol{y}(k)=\begin{bmatrix} 4\otimes x_2(k) \\ 3\otimes x_5(k) \\ 3\otimes x_7(k) \end{bmatrix} \Leftrightarrow \begin{cases} y_1(k)=4+x_2(k) \\ y_2(k)=3+x_5(k), \quad k\geq 1. \\ y_3(k)=3+x_7(k) \end{cases}$$

4. Pe baza valorilor numerice precizate în enunț se construiesc vectorii de intrare ai modelului

$$\boldsymbol{u}(1)=\begin{bmatrix}0\\0\\4\end{bmatrix},\ \boldsymbol{u}(2)=\begin{bmatrix}3\\4\\9\end{bmatrix},\ \boldsymbol{u}(3)=\begin{bmatrix}10\\4\\13\end{bmatrix},\ \boldsymbol{u}(4)=\begin{bmatrix}15\\10\\15\end{bmatrix},\ \boldsymbol{u}(5)=\begin{bmatrix}16\\20\\22\end{bmatrix},\ \boldsymbol{u}(6)=\begin{bmatrix}25\\21\\29\end{bmatrix}.$$

Utilizând formulele generale determinate la punctul 3 pentru vectorii $\boldsymbol{x}(k)$ şi $\boldsymbol{y}(k)$, $k=\overline{1,6}$, rezultă

$$\boldsymbol{x}(1)=\begin{bmatrix}0\\3\\0\\3\\7\\4\\9\end{bmatrix},\ \boldsymbol{x}(2)=\begin{bmatrix}12\\15\\9\\15\\19\\10\\17\end{bmatrix},\ \boldsymbol{x}(3)=\begin{bmatrix}20\\23\\15\\23\\27\\16\\25\end{bmatrix},\ \boldsymbol{x}(4)=\begin{bmatrix}28\\31\\21\\31\\35\\22\\33\end{bmatrix},\ \boldsymbol{x}(5)=\begin{bmatrix}36\\39\\27\\39\\43\\28\\41\end{bmatrix},\ \boldsymbol{x}(6)=\begin{bmatrix}44\\47\\33\\47\\51\\34\\49\end{bmatrix},$$

respectiv,

$$\boldsymbol{y}(1)=\begin{bmatrix}7\\10\\12\end{bmatrix},\quad \boldsymbol{y}(2)=\begin{bmatrix}19\\22\\20\end{bmatrix},\quad \boldsymbol{y}(3)=\begin{bmatrix}27\\30\\28\end{bmatrix},\quad \boldsymbol{y}(4)=\begin{bmatrix}35\\38\\36\end{bmatrix},\quad \boldsymbol{y}(1)=\begin{bmatrix}43\\46\\44\end{bmatrix},\quad \boldsymbol{y}(6)=\begin{bmatrix}51\\54\\52\end{bmatrix}.$$

5. Mediul ***Petri Net Toolbox*** furnizează atât descrierea de stare max-plus asociată rețelei din fig. AP9.4.2, cât şi reprezentarea grafică (fig. AP9.4.3) a componentelor vectorilor de intrare, stare şi ieşire menționate în enunț.

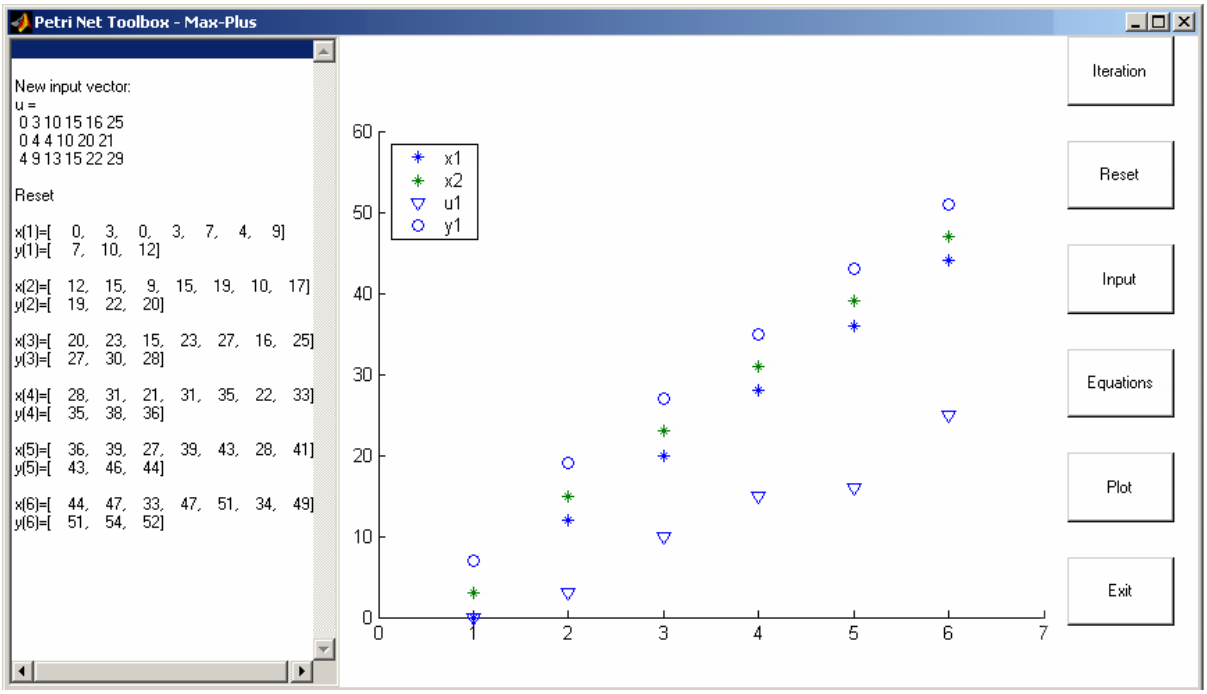


Fig. AP9.4.3. *Reprezentările grafice* $u_1(k)$, $x_1(k)$, $x_2(k)$, $y_1(k)$, *pentru* $k=\overline{1,6}$, *corespunzătoarei rețelei din fig. AP9.4.2, afişate în fereastra deschisă de* ***Petri Net Toolbox*** *pentru analiza max-plus.*

Capitolul 10

Petri Net Toolbox – descriere şi utilizare / Learning about Petri Net Toolbox

Prezentarea din acest capitol a software-ului ***Petri Net Toolbox*** este realizată în spiritul structurării din Help-ul online (http://www.ac.tuiasi.ro/pntool) pus la dispoziție de acest produs. Produsul în totalitatea lui utilizează limba engleză (pentru meniuri, cutii de dialog, documentație, etc.) cu scopul de a asigura facila lui diseminare în mediile universitare. Întrucât potențialii utilizatori aparțin unei sfere tehnico-ştiințifice bine delimitate, dezvoltată în decursul timpului pe terminologie şi concepte de origine anglo-saxonă, informațiile furnizate în cele ce urmează sunt formulate în limba engleză, considerând redundantă elaborarea unei versiuni în limba română.

Introduction – Petri Net Toolbox at a First Glance

Preface. The ***Petri Net Toolbox*** (***PN Toolbox***) is a software tool for simulation, analysis and design of discrete event systems, based on *Petri net* (PN) models. This software is embedded in the MATLAB environment and its usage requires the following products:

- **MATLAB** version 6.0 or higher;
- **Statistics Toolbox** - for random number generation and statistical functions;
- **Optimization Toolbox** - for semi-infinite minimization problems.

The integration with the MATLAB philosophy presents the considerable advantage of creating powerful algebraic, statistical and graphical instruments, which exploit the high quality routines available in MATLAB. Moreover, this MATLAB orientation of the ***PN Toolbox*** was intended to permit further development, including tools devoted to hybrid systems, because MATLAB incorporates comprehensive software for studying continuous and discontinuous dynamics.

Unlike other PN software, where places are meant as having finite capacity (because of the arithmetic representation used by the computational environment), our toolbox is able to operate with infinite-capacity places, since MATLAB includes the built-in function **Inf**, which returns the IEEE arithmetic representation for positive infinity.

The ***PN Toolbox*** was designed and implemented at the **Department of Automatic Control and Industrial Informatics** of the **Technical University „Gh. Asachi" of Iaşi**.

Expected Background. To derive a full benefit from reading this guide, you should be familiar with:

- the formal and graphical exploitation of PNs as a highly effective tool for modeling discrete event dynamical systems;
- the concepts and use of MATLAB programming language.

Types of Petri Nets. In the current version of the ***PN Toolbox***, five types of classic PN models are accepted, namely: untimed, transition timed, place-timed, stochastic and generalized stochastic. The timed nets can be deterministic or stochastic, and the stochastic case allows using the appropriate distribution function available in the MATLAB's Statistics Toolbox.

Main Features. The ***PN Toolbox*** has an easy to exploit ***Graphical User Interface*** (GUI), whose purpose is twofold. First, it gives the user the possibility to draw PNs in a natural fashion, to store, retrieve and resize (by Zoom-In and Zoom-Out features) such drawings. Second, it permits the simulation, analysis and design of the PNs, by exploiting all the computational resources of the environment, via the global variables stored in the MATLAB's Workspace. All the net nodes and arcs are handled as MATLAB objects so as changes in their parameters can be approached through the standard functions **set** and **get**.

When starting to build a new PN model, the user must define its type by appropriate checking in a dialogue box. The properties of the net-objects (places, transitions, arcs) depend on the selected type of model.

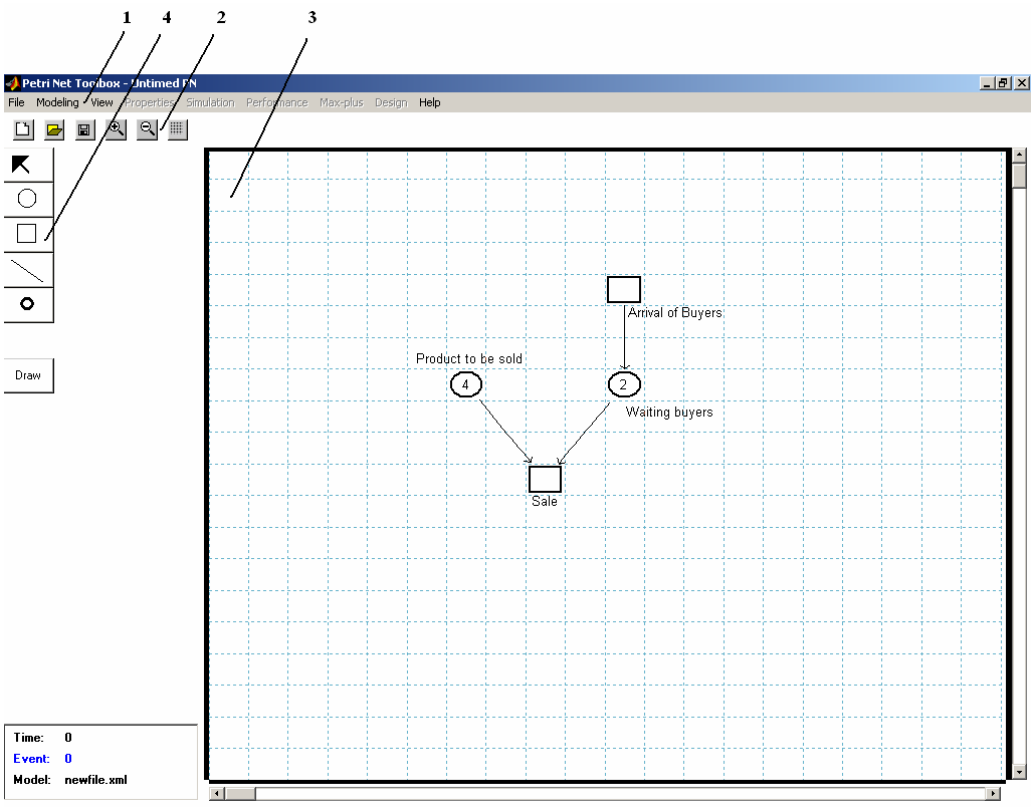
After drawing a PN model, the user can:

- visualize the ***Incidence Matrix***, which is automatically built from the net topology;
- explore the ***Behavioral Properties*** (such as liveness, boundedness, reversibility etc.) by consulting the ***Coverability Tree***, which is automatically built from the net topology and initial marking;
- explore the ***Structural Properties*** (such as structural boundedness, repetitiveness, conservativeness and consistency):
- calculate ***P-Invariants*** and ***T-Invariants***;
- run a ***Simulation*** experiment;
- display current results of the simulation using the ***Scope*** and ***Diary*** facilities;
- evaluate the global ***Performance Indices*** (such as average marking of places, average firing delay of transitions, etc.);
- perform a ***Max-Plus Analysis*** (restricted to event-graphs);
- ***Design*** a configuration with suitable dynamics (via automated iterative simulations).

The ***PN Toolbox*** uses the *XML* format for saving a model. A short description of the file format and the specific commands is given in Appendix 1 of this documentation.

To ensure flexibility, the set of all default values manipulated by the ***PN Toolbox*** are accessible to the user by means of a configuration file presented in Appendix 2.

(a)



(b)

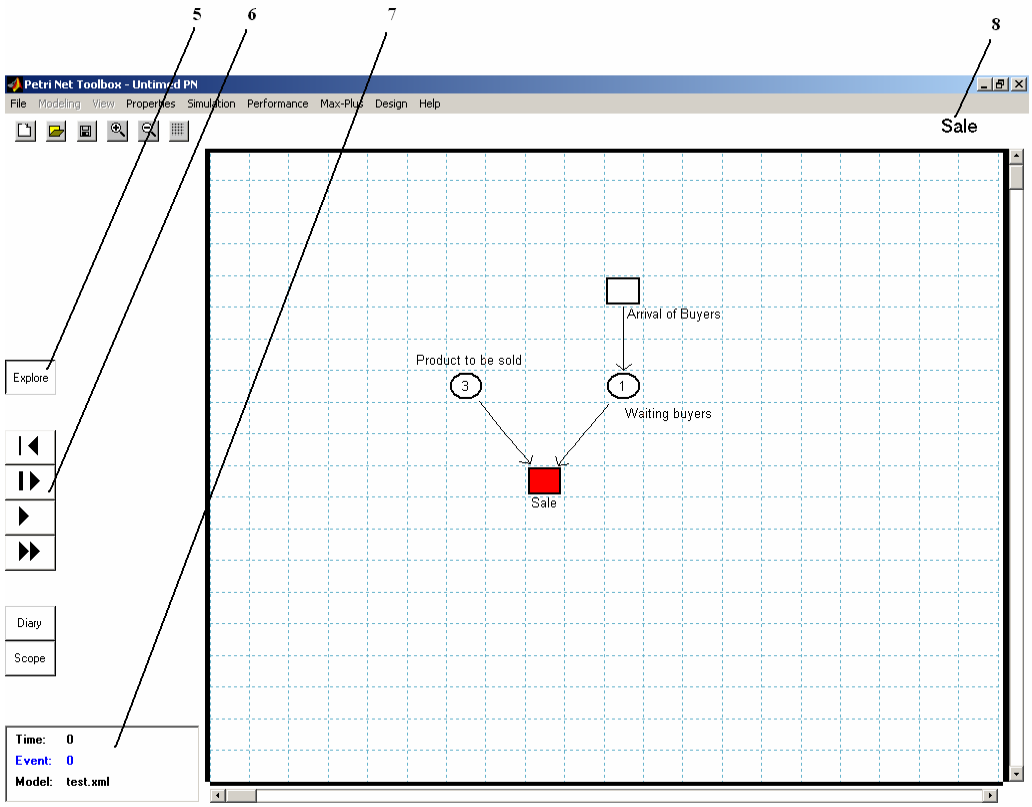


Fig. I.1. *The main **PN Toolbox** window*

(a) *Draw Mode* , (b) *Explore Mode.*

Part I. Describing the Graphical User Interface (GUI)

I.1. Overview

The ***PN Toolbox*** is equipped with an easy to exploit GUI (fig. I.1), which allows drawing PNs and permits simulation, analysis and synthesis. This GUI becomes operational once the command
>> pntool
is typed at the MATLAB's prompt.

There are two modes in which the ***PN Toolbox*** may be exploited, namely the ***Draw Mode*** that allows the user to build a new PN model or modify the properties of an existing one, and the ***Explore Mode*** that enables the user's access to simulation, analysis and design tools.

The GUI exhibits eight control panels (see fig. I.1): ***Menu Bar*** (1), ***Quick Access Toolbar*** (2), ***Drawing Area*** (3), ***Drawing Panel*** (4), ***Draw/Explore Switch*** (5), ***Simulation Panel*** (6), ***Status Panel*** (7) and a ***Message Box*** (8). Further on, all these panels are described.

I.2. Menu Bar

The ***Menu Bar*** (placed horizontally, on top of the main window of the ***PN Toolbox***) displays a set of nine drop-down menus, from which the user can access all the facilities available in the ***PN Toolbox***. These menus are enabled in accordance with the exploitation mode of the ***PN Toolbox***, namely the ***Draw Mode*** or the ***Explore Mode***.

The menus available under the ***Menu Bar*** are: ***File***, ***Modeling***, ***View***, ***Properties***, ***Simulation***, ***Performance***, ***Max-Plus***, ***Design*** and ***Help***.

I.2.1. File menu

The ***File*** menu (fig. I.2) offers facilities for file-handling operations. This is the only menu available when the ***PN Toolbox*** GUI is started. This menu contains the commands:

- ***New Model***: opens a new board in the ***Drawing Area*** for the user to build a new PN model. This command first opens a dialogue box for selecting the type of the new model;
- ***Open Model***: loads a previously saved model;
- ***Close Model***: clears the ***Drawing Area*** asking for saving the current model if changes appeared in it;
- ***Save Model***: saves the PN model drawn in the ***Drawing Area***;
- ***Save Model As …***: saves the current model with a name given by the user;
- ***Exit PNToolbox***: closes the current working session and clears all the global variables corresponding to the ***PN Toolbox***.

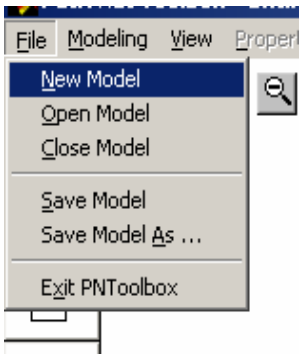


Fig. I.2. *The **File** menu.*

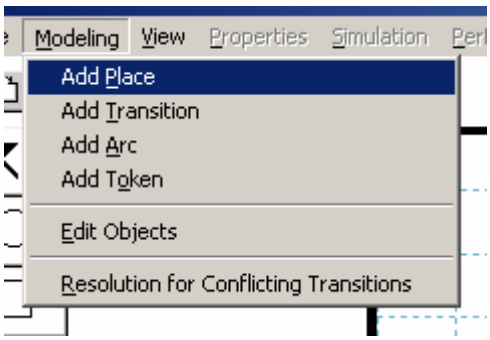


Fig. I.3. *The **Modeling** menu.*

I.2.2. Modeling menu

The ***Modeling*** menu (fig. I.3) provides tools for graphical editing (graph nodes, arcs, tokens, labels) a model in the ***Drawing Area***. The commands available in this menu are:

- ***Add Place***: adds a new place to the current PN model;
- ***Add Transition***: adds a new transition to the current PN model;
- ***Add Arc***: allows drawing an arc between two different nodes of the current PN model;
- ***Add Token***: adds a token in a specific place;
- ***Edit Objects***: opens the dialogue box associated with the net object that is selected in the ***Drawing Area*** by a click on the left button of the mouse. It allows the user to edit the properties of the selected object; these properties are in full accordance with the type of the PN model, so that part of them might be already disabled by the initial choice of the net type.
- ***Resolution for Conflicting Transitions***: allows assigning priorities or probabilities to conflicting transitions.

I.2.3. View menu

The ***View*** menu (fig. I.4) allows choosing specific conditions for visualization of the current model. The commands available in this menu are:

- ***Zoom In***: lets the user get a closer view to a smaller portion of the ***Drawing Area***;
- ***Zoom Out***: lets the user visualize a larger portion of the ***Drawing Area***;
- ***Show Grid:*** allows the user to add/remove grid lines to/from the ***Drawing Area***;
- ***Arc Weights:*** permits displaying or hiding the current values of arc weights.

Fig. I.4. *The **View** menu.*

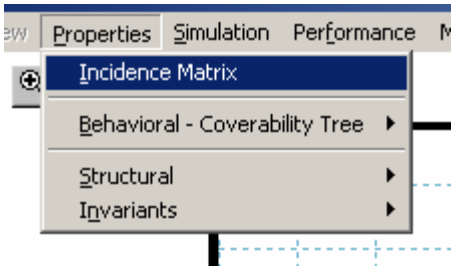


Fig. I.5. *The **Properties** menu.*

I.2.4. Properties menu

The ***Properties*** menu (fig. I.5) provides computational tools for the analysis of the behavioral and structural properties of the current PN model. The following commands are available in this menu:

- ***Incidence Matrix:*** displays the incidence matrix of the current model in a separate window;
- ***Behavioral - Coverability tree***: constructs the coverability tree of the current model, which can be visualized in two different formats: graphic and text;
- ***Structural***: performs the analysis of the structural properties of the current model (structural boundedness, conservativeness, repetitiveness and consistency);
- ***Invariants***: provides minimal-support P- and T-invariants;

I.2.5. Simulation menu

Using the ***Simulation*** menu (fig. I.6), the user can control the simulation progress and record the results (details in section II.3). The following commands are available in this menu:

- ***Step***: simulates the PN model step by step, accompanied by animation;
- ***Run Slow***: simulates the PN model with a low speed, which is specified by the user (see the ***Preferences*** command), accompanied by animation. The simulation ends when the user clicks the right button of the mouse;
- ***Run Fast***: simulates the PN model with the maximum speed allowed by the environment. No animation is provided in this case. The simulation ends when the condition set by the user in ***Breakpoint*** is met;
- ***Breakpoint***: allows choosing the stop condition for simulation in the ***Run Fast*** mode;
- ***Reset***: brings the PN model to the initial state and resets all the results that were recorded in a previous simulation experiment;

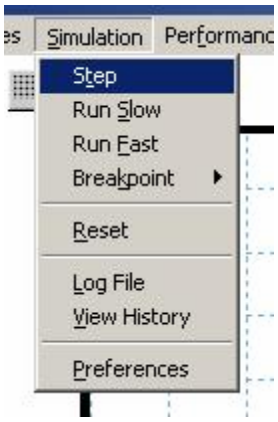


Fig. I.6. *The* ***Simulation*** *menu.*

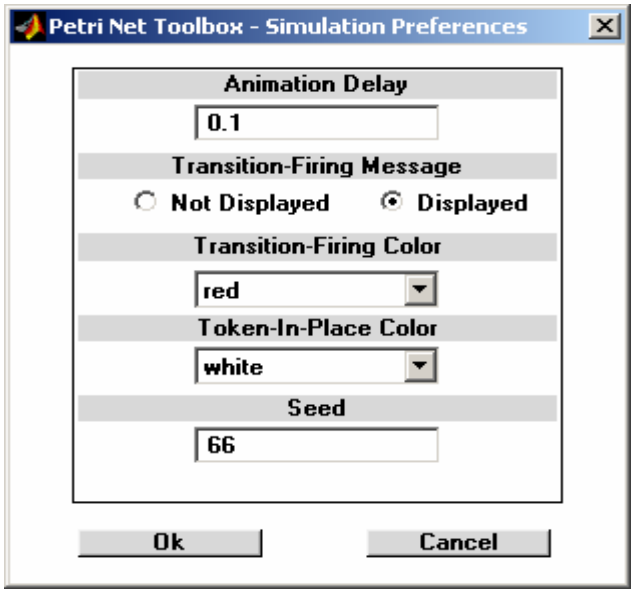


Fig. I.7. *The* ***Preferences*** *dialogue box.*

- ***Log File***: writes the simulation history (recording all the states that the PN model passed through) in a file designated by the user. This command is operative only in the ***Step*** and ***Run Slow*** modes;
- ***View History***: opens the simulation log file;
- ***Preferences***: opens a dialogue box (fig. I.7) allowing the user to set the conditions of the simulation (***Animation Delay*** is valid only for the ***Run Slow*** simulation mode, ***Token-In-Place Color*** is meaningful only for P-timed PNs, the value of the ***Seed*** is used to initialize the random number generator of MATLAB).

I.2.6. Performance menu

At the end of a simulation experiment, the ***Performance*** menu (fig. I.8) allows the visualization of the ***global performance indices*** that are stored in an HTML format. These indices are separately recorded for transitions and for places. The commands available for this menu are:

- ***Place Indices***: displays the performance indices corresponding to the places;
- ***Transition Indices***: displays the performance indices corresponding to the transitions.

Fig. I.8. *The **Performance** menu*.

Fig. I.9. *The **Help** menu*.

I.2.7. Max-Plus menu

The ***Max-Plus*** menu allows performing the simulation and analysis of an event graph (marked graph) based on its max-plus state-space model. A new MATLAB figure is opened and all the facilities available for max-plus analysis and simulation are accessible (see section **Max-Plus Models**).

I.2.8. Design menu

The ***Design*** menu is used for the synthesis of timed PN models; this allows simulations for several types of parameterizations considered in the PN architecture (see section II.6).

I.2.9. Help menu

The ***Help*** menu (fig. I.9) provides information about the exploitation of the ***PN Toolbox***. The commands available in this menu are as follows:

- ***Contents and Index***: opens this documentation;
- ***Demos***: allows visualization of four **Flash** movies initiating the user in the exploitation of the ***PN Toolbox***;
- ***About Petri Net Toolbox***: contains information about the authors of the ***PN Toolbox***.

I.3. Quick Access Toolbar

The ***Quick Access Toolbar*** (fig. I.10) is placed as a horizontal bar just below the ***Menu Bar*** and presents six image buttons. The actions of the first three buttons are identical to those controllable by the ***New*** (1), ***Open*** (2) and ***Save*** (3) commands from the ***File*** menu. The actions of the next three buttons are identical to those controllable by the ***Zoom In*** (4), ***Zoom Out*** (5) and ***Show Grid*** (6) commands from the ***View*** menu.

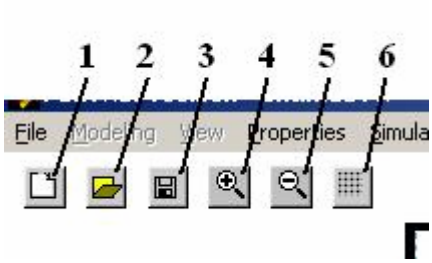


Fig. I.10. *The* ***Quick Access Toolbar****.*

I.4. Drawing Area

The ***Drawing Area*** (located in the central and right side of the main window – see fig. I.1) is provided with a grid, where the nodes of the PN graph are to be placed, and with two scrollbars (on the right and bottom sides) for moving the desired parts of the graph into view.

The ***Drawing Area*** is an axes MATLAB object and it is organized as a matrix with 50 rows and 50 columns. The bottom left-hand corner corresponds to the (0,0) cell, whereas the upper right-hand corner to the (49,49) cell. In one cell the user can draw a single node (place or transition), so that the total number of places and transitions in a model cannot exceed 2500.

Using the ***Zoom In*** / ***Zoom Out*** commands from the ***View*** menu, the number of the cells is decreased/increased with 10 cells on each axis. The ***Show Grid*** command adds/removes the grid lines to/from the Drawing Area.

I.5. Drawing Panel

The ***Drawing Panel*** (fig. I.11) is placed vertically, in the left side of the main window, below the ***Quick Access Toolbar***, and presents five image buttons with actions identical to the commands ***Edit Objects*** (1), ***Add Place*** (2), ***Add Transition*** (3), ***Add Arc*** (4) and ***Add Token*** (5) available in the ***Modeling*** menu.

I.6. Draw/Explore Switch

The ***Draw / Explore Switch*** (identified as (5) in fig. I.1) allows switching between the ***Draw Mode*** (in which the user can draw a new model or modify an existing one – button unpressed) and the ***Explore Mode*** (in which the user can access all functions available for simulation and analysis – button pressed).

When switching from the ***Explore Mode*** to the ***Draw Mode***, the marking of the PN model is automatically reset to the initial token configuration and the former values of all the performance indices are lost.

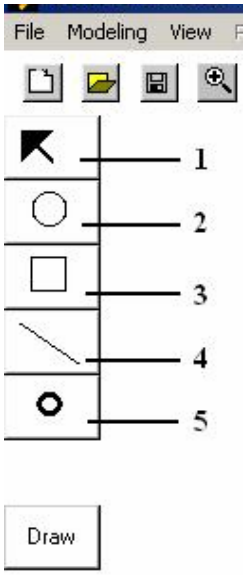


Fig. I.11. *The **Drawing Panel***.

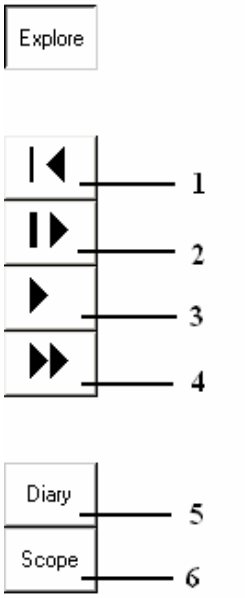


Fig. I.12. *The **Simulation Panel***.

I.7. Simulation Panel

The ***Simulation Panel*** (fig. I.12) is placed vertically, in the left side of the main window, just below the ***Drawing Panel***. It presents four image buttons with actions identical to the commands ***Reset*** (1), ***Step*** (2), ***Run Slow*** (3) and ***Run Fast*** (4) available in the ***Simulation*** menu. It also provides two visualization instruments for observing the progress of the simulation: ***Diary*** (5) and ***Scope*** (6) (see section **Running a Simulation**).

I.8. Status Panel

The ***Status Panel*** (identified as (7) in fig. I.1) is a message board (placed in the bottom left-hand corner of the main window), where the ***PN Toolbox*** displays the current simulation time and the total number of events. In addition, the ***Status Panel*** displays the file name corresponding to the current model.

I.9. Message Box

The ***Message Box*** (identified as (8) in fig. I.1) is a MATLAB text object used by the ***PN Toolbox*** to display messages to the user. In the ***Draw Mode***, five types of messages may be displayed, corresponding to the actions enabled by the five buttons in the ***Drawing Panel***, respectively. In the ***Explore Mode***, messages are displayed only during the simulation of a model (***Step*** or ***Run Slow*** commands) if option ***Displayed*** is checked for ***Transition-Firing Message*** in the ***Preferences*** dialogue box (from the ***Simulation*** menu). When a transition fires, the ***Message Box*** displays a text associated with this transition; the text must be previously defined in the ***Edit Transition*** dialogue box (***Draw Mode*** - see section II.1.3).

Part II. Exploiting the Toolbox

II.1. Building a Model

II.1.1. Overview

The ***PN Toolbox*** provides a set of commands for building a PN model, which are accessible from the ***Modeling*** menu or from the corresponding buttons in the ***Drawing Panel***. The model may be easily drawn, in a natural fashion, in the ***Drawing Area***. The user can also have a read / write access to the properties of the net nodes and arcs by opening the dialog box associated with the object selected in the ***Drawing Area***. In addition, the ***PN Toolbox*** allows the assignment of priorities and / or probabilities to conflicting transitions.

The ***Drawing Area*** displays a grid with dotted gray lines. The user may hide or show these lines by using the command ***View / Show Grid*** or by pressing the corresponding button from the ***Quick Access Toolbar***. The nodes of the PN may be placed only in the grid cells; each cell can contain a single net node. This way, each node is uniquely characterized by its coordinates in the ***Drawing Area***.

A PN model created in the ***PN Toolbox*** is saved as an *XML* file. Besides the automatic generation of the XML file of a model, any text editor can be used to produce such a file if the user complies with the syntax given in Appendix 1.

The first step in the creation of a new model is to define its type. By selecting the type of the PN, some of the work variables are automatically changed. This is because the simulation and analysis procedures differ from one type to another. The drawing board of a new model may be opened in the ***Drawing Area*** by selecting the ***File / New Model*** command or by pressing the ***New*** button from ***Quick Access Toolbar***. This command opens a dialog box (fig. II.1) which permits the selection of the type of model to be built. The default type is ***Untimed PN***.

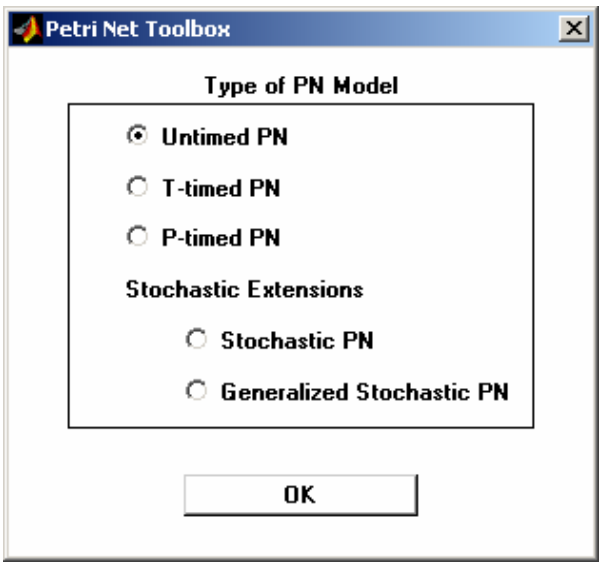


Fig. II.1. *The dialogue box for selecting the type of a PN model.*

A previously created model may be loaded from the disk by means of the command ***Open*** available under the ***File*** menu or by pressing the ***Open*** button from the ***Quick Access Toolbar***. Changes in an existing model can be done directly in the GUI while in ***Draw Mode***, or by editing the corresponding **XML** file.

If a model is already loaded in the window, the ***Drawing Area*** is cleared by selecting the command ***File / Close Model***.

II.1.2. Places

A place is graphically represented in the ***Drawing Area*** by a circle. To draw a place in the ***Drawing Area***, the user must press the ***Add Place*** button from the ***Drawing Panel*** or select the ***Add Place*** command from the ***Modeling*** menu. Then, the user must click only once into the desired grid cell of the ***Drawing Area***. A second click in the same grid cell has no effect.
Once the circle corresponding to a place is drawn, a label is automatically attached to it. The default position of the label is below the circle.

After drawing a place, there are two ways to change its position in the ***Drawing Area*** while in ***Draw Mode*** and no button from the ***Drawing Panel*** is pressed. The first way is to left-click the desired place and then drag it in the new position. The second way is to right-click the place; as a result, a MATLAB uicontext menu (fig. II.2) appears and the command ***Move Place*** becomes available. In both cases, the label is moved together with the place.

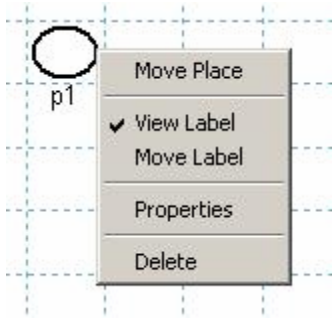


Fig. II.2. *The uicontext menu of a place.*

For a selected place, the uicontext menu also allows: (*i*) controlling the label's visibility on the screen (***View Label*** command), (*ii*) changing the position of the label (***Move Label*** command), (*iii*) deleting the place (***Delete*** command) and (*iv*) opening the ***Edit Place*** dialogue box (fig. II.3) that lets the user modify the properties of the place as a MATLAB object (***Properties*** command).

The ***Edit Place*** dialogue box may be also opened by one of the procedures (EP1) or (EP2) described below, followed by a click on the desired place. (EP1) consists in selecting the ***Edit Objects*** command from the ***Modeling*** menu; (EP2) consists in pressing the ***Edit Objects*** button from the ***Drawing Panel***.

Each place is uniquely identified with an id that is automatically assigned by the ***PN Toolbox*** and cannot be changed by the user. This id appears in the title bar of the ***Edit Place*** dialogue box.

The option ***Name*** displays the string that is used as the label of the place. By default, this string coincides with the id of the place. The user can modify this string (without affecting the id) if necessary.

The option ***Color*** displays the color used for drawing the place. By default, this color is black, but the user can select another one from a list of eight predefined colors. Using different colors for drawing the places of a net might be helpful for complex topologies (e.g. for highlighting different phases requested by a multi-step design procedure).

The option ***Capacity*** displays the capacity of the place. By default, the capacity is **Inf** (the IEEE arithmetic representation for positive infinity). The user can set this field to a positive integer value.

The option ***Tokens*** displays the number of tokens in the place. By default, this number is 0. The user can set this field to a positive integer value. If the marking of a place is greater than 0, this marking is shown as a number inside the circle corresponding to that position. Void marking is not explicitly shown.

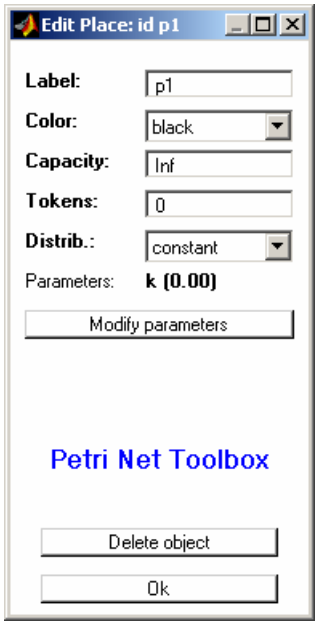


Fig. II.3. *The **Edit Place** dialogue box for modifying the properties of a place.*

There are two more possibilities to add a token to a place (AT1) or (AT2) described below, followed by a click on the desired place. (AT1) consists in selecting the ***Add Token*** command from the ***Modeling*** menu; (AT2) consists in pressing the ***Add Token*** button from the ***Drawing Panel***.

In case of place-timed PN models (see section II.3.4), the ***Distribution*** option associated with a place allows the user to specify the probability distribution and the necessary parameter(s) which define the corresponding time-duration. This option is not available for untimed or transition-timed models.

The ***Delete object*** button placed at the bottom of the ***Edit Place*** dialogue box lets the user delete the place from the model.

II.1.3. Transitions

A transition is graphically represented in the ***Drawing Area*** by a square. To draw a transition in the ***Drawing Area***, the user must press the ***Add Transition*** button from the ***Drawing Panel*** or select the ***Add Transition*** command from the ***Modeling*** menu. Then, the user must click only once into the desired grid cell of the ***Drawing Area***. A second click in the same grid cell has no effect.

Once the square corresponding to a transition is drawn, a label is automatically attached to it. The default position of the label is below the square.

The same as for a place, after drawing a transition, there are two ways to change its

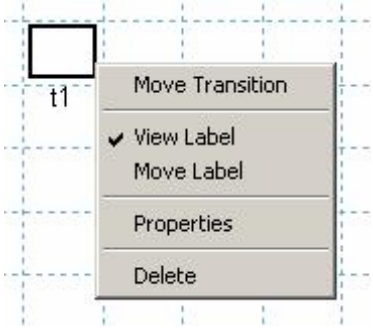


Fig. II.4. *The uicontext menu of a transition.*

position in the ***Drawing Area*** while in ***Draw Mode*** and no button from the ***Drawing Panel*** is pressed. The first way is to left-click the desired transition and then drag it in the new position. The second way is to right-click the transition; as a result, a MATLAB uicontext menu (fig. II.4) appears and the command ***Move Transition*** becomes available. In both cases, the label is moved together with the transition.

For a selected transition, the uicontext menu also allows: (*i*) controlling the label's visibility on the screen (***View Label*** command), (*ii*) changing the position of the label (***Move Label*** command), (*iii*) deleting the transition (***Delete*** command) and (*iv*) opening the ***Edit Transition*** dialogue box (fig. II.5) that lets the user modify the properties of the transition as a MATLAB object (***Properties*** command).

The ***Edit Transition*** dialogue box can be also opened by one of the procedures (ET1) or (ET2) described below, followed by a click on the desired transition. (ET1) consists in selecting the ***Edit Objects*** command from the ***Modeling*** menu; (ET2) consists in pressing the ***Edit Objects*** button from the ***Drawing Panel***.

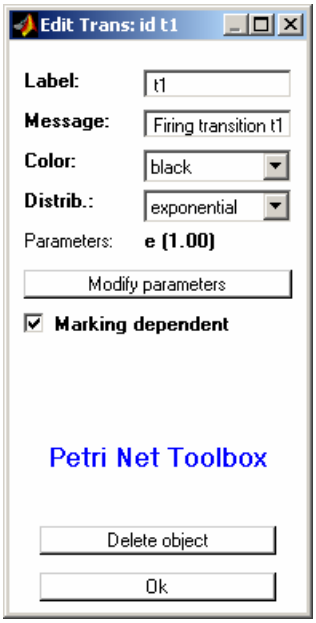


Fig. II.5. *The **Edit Transition** dialogue box for modifying the properties of a transition*

Each transition is uniquely identified with an id that is automatically assigned by the ***PN Toolbox*** and cannot be changed by the user. This id appears in the title bar of the ***Edit Transition*** dialogue box.

The option ***Name*** displays the string that is used as the label of the transition. By default, this string coincides with the id of the transition. The user can modify this string (without affecting the id) if necessary.

The ***Message*** text-box contains the string that is displayed in the ***Message Box*** during the simulation of the model when firing that transition if option ***Displayed*** is checked for ***Transition-Firing Message*** in the ***Preferences*** dialogue box. By default, this string is "*Firing transition x*" where *x* is the id of that transition. The user can modify this string if necessary.

The option ***Color*** displays the color used for drawing the transition. By default, this color is black, but the user can select another one from a list of eight predefined colors. Using different colors for the transitions of a net might be helpful for complex topologies (e.g. for highlighting different phases requested by a multi-step design procedure).

In case of transition-timed PN models (see section II.3.3), the ***Distribution*** option associated with a transition allows the user to specify the probability distribution and the necessary parameter(s) which define the corresponding time-duration. This option is not available for untimed or place-timed models.

For Stochastic PN models, only exponentially distributed firing rates can be assigned to the transitions in the net. In the case of Generalized Stochastic PN models, only constant and exponentially distributed firing rates can be assigned to the transitions in the net;

obviously, an instantaneously fireable transition can be modeled by setting the constant distribution to 0. For these two types of models, the user has the possibility to enable or disable the dependence between the firing rate of a transition and the marking of its input places by checking the ***Marking dependent*** option.

The ***Delete object*** button placed at the bottom of the ***Edit Place*** dialogue box lets the user delete the transition from the model.

II.1.4. Arcs

To draw an arc in the ***Drawing Area***, the user must press the ***Add Arc*** button from the ***Drawing Panel*** or select the ***Add Arc*** command from the ***Modeling*** menu. Then, the user must click on the start node and then on the end node. The implementation in the ***PN Toolbox*** of PN models complies with the basic rule: *an arc of a PN can only connect a place to a transition* (pre-arc) *or a transition to a place* (post-arc)*, but never two nodes of the same kind*. The role of splitting arcs in two categories (pre- and post-arcs) will become apparent below, when talking about inhibitor arcs.

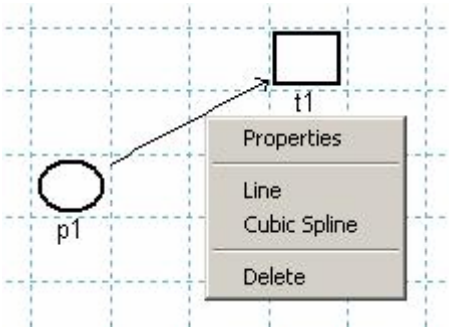


Fig. II.6. *The uicontext menu of an arc.*

By default, an arc is represented as a straight arrow between the two selected nodes of the net. While in ***Draw Mode*** and no button from the ***Drawing Panel*** is pressed, a right-click on an arc of the net opens a MATLAB uicontext menu (fig. II.6) that allows: (*i*) modifying the graphical representation in the ***Drawing Area*** (***Line*** command for a straight line or ***Cubic Spline*** command for a curve), (*ii*) deleting the arc (***Delete*** command) and (*iii*) opening the ***Edit Arc*** dialogue box (fig. II.7) that lets the user modify the properties of the arc as a MATLAB object (***Properties*** command).

The ***Edit Arc*** dialogue box can be also opened by one of the procedures (EA1) or (EA2) described below, followed by a click on the desired arc. (EA1) consists in selecting the ***Edit Objects*** command from the ***Modeling*** menu; (EA2) consists in pressing the ***Edit Objects*** button from the ***Drawing Panel***.

Each arc is uniquely identified with an id that is automatically assigned by the ***PN Toolbox*** and cannot be changed by the user. This id appears in the title bar of the ***Edit Arc*** dialogue box.

The option ***Color*** displays the color used for drawing the arc. By default, this color is black, but the user can select another one from a list of eight predefined

Fig. II.7. *The **Edit Arc** dialogue box for modifying the properties of an arc.*

colors. Using different colors for the arcs of a net might be helpful for complex topologies (e.g. for highlighting different phases requested by a multi-step design procedure).

The option ***Type*** displays the type of an arc. By default, the type of an arc of a PN model is regular, but the user can change it into bidirectional or inhibitor, if necessary. A *regular* arc represents the standard connection between two nodes of different types. A *bidirectional* arc is equivalent to a pair of arcs with the same weight, one connecting a place to a transition and the other one connecting the same transition to the same place. The graphical representation of a bidirectional arc is a line with arrows at both ends.

An *inhibitor* arc can connect only a place to a transition. The transition is enabled only if the number of tokens in the input place is strictly smaller than the weight of the inhibitor arc. The graphical representation of an inhibitor arc is a line between the two nodes ending with a small circle (near the inhibited transition).

The option ***Weight*** displays the weight (multiplicity) of the arc. By default, the weight of an arc is equal to 1, but the user can set this field to a positive integer value.

By default, the weights of the arcs in a net are not shown in the ***Drawing Area***. At any stage of PN drawing, the user can visualize the current values of the weights for all the arcs in the net by selecting the ***Arc Weights*** command from the ***View*** menu. This command opens the dialogue box presented in fig. II.8 and the user must click the ***Yes*** button.

Fig. II.8. *The **View Arc Weights** dialogue box.*

II.1.5. Setting Priorities and / or Probabilities for Conflicting Transitions

By default, the ***PN Toolbox*** sets equal probabilities and / or priorities to conflicting transitions. These probabilities and / or priorities are used by the ***PN Toolbox***, when simulating a model, for selecting, from a set of conflicting transitions enabled at the same time, the next one to be fired. By selecting the ***Conflicting Transitions*** option available under the ***Modeling*** menu, the corresponding dialogue window is opened (fig. II. 9) and the user can add, delete or edit the probabilities and / or priorities for each group of conflicting transitions.

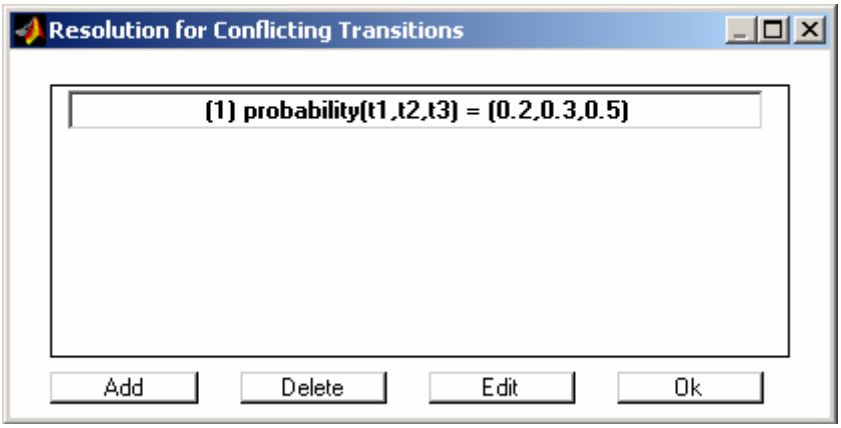


Fig. II.9. *The **Resolution for Conflicting Transitions** dialogue window.*

II.2. Exploring Properties

II.2.1. Overview

The ***PN Toolbox*** offers several possibilities for analyzing discrete event systems modeled with PNs that are briefly presented in the sequel.

II.2.2. Incidence Matrix

The incidence matrix of a PN is computed directly from the graphical model and can be visualized in a MATLAB window opened by selecting the ***Incidence Matrix*** command from the ***Properties*** menu. This command is available for all types of PNs.

II.2.3. Behavioral Properties

For untimed PN models, the behavioral properties (e.g. boundedness, liveness, reversibility, etc.) may be studied based on the coverability tree of the net. Relying on the topology and the initial marking of the net, the ***PN Toolbox*** can automatically construct this tree and display it in either text or graphical mode. The coverability tree is built with or without the ω-convention (fig. II.10). The ω-convention means the usage of a generic symbol (herein denoted by "ω") for referring to unbounded markings (Murata, 1989). Since the boundedness of a net cannot be a priori known, it is recommended to start the construction of the coverability tree by answering ***No*** to the question ***Do you want to use the "ω - convention"?***.

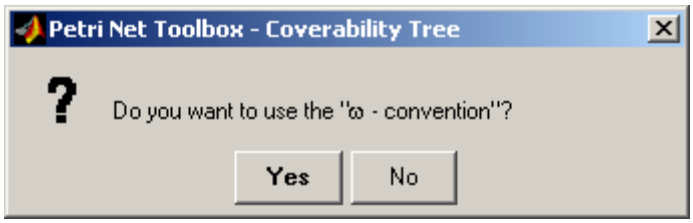


Fig II.10. *The **Coverability Tree** dialogue window.*

II.2.4. Structural Properties

The *structural properties* of a PN are independent of the initial marking and may be characterized in terms of linear inequalities based on the incidence matrix of the net. The structural properties can be investigated from the ***Properties / Structural Properties*** menu, by means of the ***Structural Boundedness***, ***Conservativeness***, ***Repetitiveness*** and ***Consistency*** options.

II.2.5. Invariants

From the ***Properties / Invariants*** menu, the minimal-support *P-* and *T-invariants* (Martinez and Silva, 1982) can be visualized as vectors displayed in separate windows, automatically opened by the ***PN Toolbox***. New invariants may be constructed as linear combinations of these vectors.

II.3. Running a Simulation

II.3.1. Overview

After a model was drawn or retrieved from the disk, the user can simulate it. Three modes of simulation are implemented in the ***PN Toolbox***, namely: ***Step***, ***Run Slow*** and ***Run Fast***, that are called from the ***Simulation*** menu or from the ***Simulation Panel***.

The ***Step*** simulation mode ensures the progress of the simulation by a single step, when the status of a transition is changed from enabled to fired, or from fired to disabled. With the occurrence of each event two successive steps are associated, meaning the complete treatment of the firing transition.

The ***Step*** and ***Run Slow*** simulation modes are accompanied by animation. The animation displays the current number of tokens in the places of the net and colors transitions when they are fired. In the ***Run Slow*** mode, the speed of animation is adjustable from the ***Animation Speed*** option in the ***Simulation / Preferences*** dialogue box. The simulation can be stopped with a right-click.

In these two modes, the ***Status Panel*** (placed in the bottom left-hand corner of the main window) presents the number of events and the current time.

Also, the user can record the progress of the simulation by using the ***Log File*** command from the ***Simulation*** menu. This journal is automatically saved in the project directory as an *HTML* file and can be read by means of any Internet browser using the ***Simulation / View history*** command available in the ***PN Toolbox***.

Note that in ***Step*** or ***Run Slow*** modes, the ***Scope*** and ***Diary*** facilities are available during the simulation. Their usage is relevant when studying timed PN.

The ***Run Fast*** simulation mode is not accompanied by animation and the simulation speed depends on the computer performances. This mode is recommended when the user is interested in estimating global performance indices of the PN model over a large time horizon (or, equivalently, for the occurrence of a large number of events).

From the ***Simulation / Breakpoint*** submenu the user can control the end point of the simulation in the ***Run Fast*** mode. The simulation is stopped by default when 1000 events have occurred or if no transition can fire any longer (i.e. when deadlock appears).

In all simulation modes, immediately after the end of the simulation, the values of the Performance Indices which characterize the simulated dynamics are available on request, by accessing the menu ***Performance***.

The simulation principle is different from one type of PN to another. Details about the simulation method implemented in the ***PN Toolbox*** for each type of PN model are given in the next pages.

II.3.2. Untimed Petri Nets

The sequencing of the events is reduced to simply ordering their occurrences. The simulation runs by firing the transitions one-by-one, in accordance with the *transition firing rule*. The ***PN Toolbox*** stores all the enabled transitions in a MATLAB array but only one transition fires at a time. The function that returns the next transition to be fired makes the decision according to

the priorities or probabilities assigned to conflicting transitions from the ***Modeling / Conflicting Transitions*** command.

For the simulation modes ***Step*** and ***Run Slow***, the ***Events*** counter in the ***Status Panel*** is indexing and the currently firing transition gets the red color (by default) or an arbitrary color (selected by the user from the option ***Simulation / Preferences***). After it is fired, a transition returns to the background color.

The usage of the ***Scope*** and ***Diary*** facilities is not relevant for this type of PN.

II.3.3. T-timed Petri Nets

For *transition-timed* PN (*T*-timed PN), time durations can be assigned to the transitions; tokens are meant to spend that time as reserved in the input places of the corresponding transitions. In simulation, all the transitions that can fire due to the current marking are fired at the same time. For conflicting transitions, priorities or probabilities allow the choice of the transition(s) to fire. A transition can fire several times, in accordance with the marking of its input places and, from a theoretical point of view, an infinitesimal delay is considered to separate any two successive firings. After a transition fires, the enabling condition is tested for all the transitions of the net.

For the time durations assigned to the transitions, the appropriate MATLAB functions available in the Statistics Toolbox can be used. When a transition is waiting to fire (i.e. during the period when its input places contain reserved tokens) the reserved tokens are graphically removed from the input places, but the computation procedures of the performance indices consider them as remaining in those places (in full accordance with the theoretic approach).

The ***Diary*** facility opens a new window that displays (dynamically) the instant(s) when the next fireable transition(s) is (are) to fire (see section II.3.7).

The ***Scope*** facility opens a new figure window that displays (dynamically) the evolution of the selected performance indices. The final value of the global index shown by the Scope is identical to that displayed on request, at the end of simulation, when the user explores the ***Performance*** menu (see section I.2.6).

II.3.4. P-timed Petri Nets

For *place-timed* PN (*P*-timed PN), time durations can be assigned to the places; tokens are meant to spend that time as reserved in the corresponding places, immediately after their arrival. In simulation, all the transitions that can fire due to the current marking, fire at the same time. A transition can fire several times, in accordance with the marking of its input places and, from a theoretical point of view, an infinitesimal delay is considered to separate any two successive firings.

For the time durations assigned to the places, the appropriate MATLAB functions available in the Statistics Toolbox can be used. During the simulation, the places that contain reserved tokens get the white color (default) or a color selected by using the option ***Simulation / Preferences***.

The ***Diary*** facility opens a new window that displays (dynamically) the instant(s) when the next fireable transition(s) is (are) to fire (see section II.3.7).

The ***Scope*** facility opens a new figure window that displays (dynamically) the evolution of the selected performance indices. The final value of the global index shown by the Scope is identical to that displayed on request, at the end of simulation, when the user explores the ***Performance*** menu (see section I.2.6).

II.3.5. Stochastic Petri Nets

For *stochastic* PNs (SPNs), only exponential type distributions can be used to assign the time durations of the transitions. For conflicting transitions, it is the shortest time duration that allows the choice of the transition to fire, without using priorities or probabilities. Multiple firing of the same transition is not permitted, even if the token content of its input places allows this; i.e. the transition fires once and after the allocated time elapses, it will fire again if the current marking is appropriate. (Notice that the mechanism for selecting the firing transition and the condition for firing only once make the difference from the *T-timed PNs* with exponential type distributions for the time durations).

The sequencing of the firing transitions is exclusively controlled by the time durations of exponential type, which ensures the equivalence with the Markov chains. The firing rate of the transitions (i.e. the inverse of the mean time-duration) is, by default, marking dependent, but the user can select a marking-independent operation.

The ***Diary*** facility opens a new window that displays (dynamically) the instant(s) when the next fireable transition(s) is (are) to fire.

The ***Scope*** facility opens a new figure window that displays (dynamically) the evolution of the selected performance indices. The final value of the global index shown by the Scope is identical to that displayed on request, at the end of simulation, when the user explores the ***Performance*** menu (see section I.2.6).

II.3.6. Generalized Stochastic Petri Nets

Generalized Stochastic PNs (GSPNs) have two different classes of transitions: *immediate* transitions and *timed* transitions. Once enabled, immediate transitions fire in zero time. Timed transitions fire after a random, exponentially distributed enabling time as in the case of SPNs. For *timed* transitions, the firing rate (i.e. the inverse of the mean time-duration) is, by default, marking dependent, but the user can select a marking-independent operation (the same way as for SPNs).

The simulation procedure is similar to the SPN case, the only difference occurring in the case of the *immediate* transitions that fire first; priorities / probabilities can be associated to these transitions, in order to resolve the conflicts.

The ***Diary*** facility opens a new window that displays (dynamically) the instant(s) when the next fireable transition(s) is (are) to fire.

The ***Scope*** facility opens a new figure window that displays (dynamically) the evolution of the selected performance indices. The final value of the global index shown by the Scope is identical to that displayed on request, at the end of simulation, when the user explores the ***Performance*** menu (see section I.2.6).

II.3.7. Diary

When active, this facility opens a new MATLAB window (fig. II.11) that displays (dynamically - during the simulation) the instant(s) when the next fireable transition(s) is (are) to fire. For several fireable transitions, the displayed list of transitions is ordered by the instants when the firings will occur.

The ***Diary*** facility is available only in the ***Step*** and ***Run Slow*** simulation modes. The usage of this facility is recommended for timed PN models and it gives no relevant information in the case of untimed models.

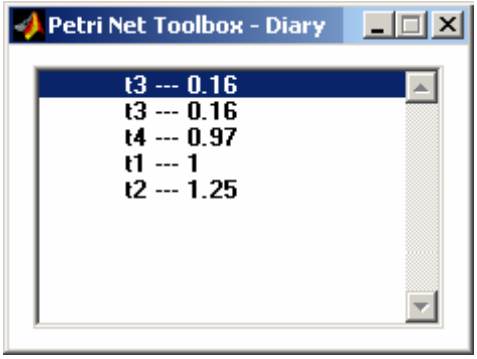


Fig II.11. *The **Diary** window.*

II.3.8. Scope

The ***Scope*** facility opens a new MATLAB window (fig. II.12) that displays (dynamically) the evolution of the selected performance index. Each index provides two types of information, namely a current value (that characterizes the PN at the current moment of the simulation) and a global value (that characterizes the whole evolution of the PN, as an average over the whole time-horizon from the beginning of the simulation till the current moment). For the selected performance index, both current and global values are plotted versus time.

The usage of the ***Scope*** facility is available only for timed PNs in the ***Step*** and ***Run Slow*** simulation modes.

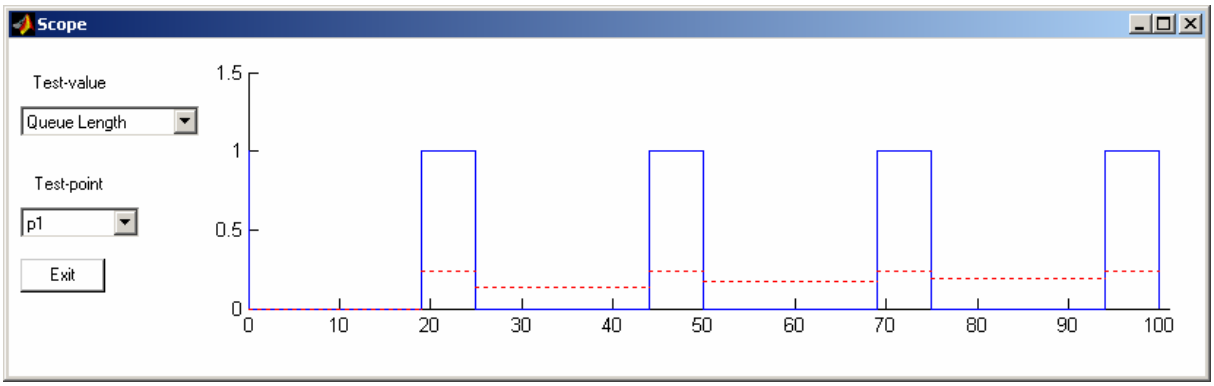


Fig II.12. *The **Scope** window.*

The current / global performance indices displayed by ***Scope*** are:

- for a transition:
  - ***Service Distance***: means the duration between two successive firings of the selected transition;
  - ***Service Time***: means the duration between the enabling and the firing of the selected transition;

- ***Utilization***: means the current status of the selected transition: 1 for the time-interval between the enabling and the firing of the selected transition; 0 otherwise;

- for a place:
  - ***Arrival Distance***: means the duration between two successive instants when tokens arrive in the selected place;
  - ***Throughput Distance***: means the duration between two successive instants when tokens leave the selected place;
  - ***Queue Length***: means the number of tokens in the selected place.

During a simulation experiment, only one of the six performance indices can be dynamically displayed by the ***Scope***. The red color is automatically set for plotting the evolution of the global performance index. The blue color is automatically set for plotting the evolution of the current performance index. The final value shown by the ***Scope*** for the global performance index is identical to the value displayed on request, at the end of simulation, by accessing the ***Performance*** menu.

II.4. Analyzing Simulation Results

After simulation ends, the global performance indices described in the ***Scope*** section are stored by the ***PN Toolbox*** and can be visualized by using the ***Performance*** menu. Besides these, there are also a number of global indices for which the current values are not defined.

The following two tables present the complete lists of global indices associated with the places (displayed by the ***Place Indices*** command) and the transitions (displayed by the ***Transition Indices*** command), respectively:

- for a transition:
  - ***Service Sum***: total number of firings;
  - ***Service Distance***: average value of the current index ***Service Distance***;
  - ***Service Rate***: average frequency of firings (inverse of ***Service Distance***);
  - ***Service Time***: average value of the current index ***Service Time***;
  - ***Utilization***: average value of the current index ***Utilization***;
- for a place:
  - ***Arrival Sum***: total number of arrived tokens;
  - ***Arrival Distance***: average value of the current index ***Arrival Distance***;
  - ***Arrival Rate***: average frequency of token-arrivals (inverse of ***Arrival Distance***);
  - ***Throughput Sum***: total number of departed tokens;
  - ***Throughput Distance***: average value of the current index ***Throughput Distance***;
  - ***Throughput Rate***: average frequency of token-departures (inverse of ***Throughput Distance***);
  - ***Waiting Time***: average waiting time per token;
  - ***Queue Length***: average value of the current index ***Queue Length***.

These indices may be saved in *HTML* format in a file placed in the working directory.

II.5. Max-Plus Models

For place-timed event graphs, the ***PN Toolbox*** is able to directly derive the max-plus state-space representation from the topology and initial marking of a marked graph, in an implicit form:

$$\begin{aligned} \boldsymbol{x}(k) &= \bigoplus_{i=0}^{M} \left[\boldsymbol{A}_i \otimes \boldsymbol{x}(k-i) \oplus \boldsymbol{B}_i \otimes \boldsymbol{u}(k-i) \right], \\ \boldsymbol{y}(k) &= \bigoplus_{i=0}^{M} \left[\boldsymbol{C}_i \otimes \boldsymbol{x}(k-i) \oplus \boldsymbol{D}_i \otimes \boldsymbol{u}(k-i) \right], \end{aligned} \quad , \quad k = \overline{1, N},$$

where M denotes the maximal number of tokens in the initial marking and N stands for the number of simulated iterations. The components of the input vector $\boldsymbol{u}(k) = \left[u_1(k)\ u_2(k)\ \ldots\ u_m(k) \right]^{\mathrm{T}}$ and those of the output vector $\boldsymbol{y}(k) = \left[y_1(k)\ y_2(k)\ \ldots\ y_p(k) \right]^{\mathrm{T}}$ represent the k-th firing moments of the m source transitions and of the p sink transitions, respectively. In a similar manner, the state vector $\boldsymbol{x}(k) = \left[x_1(k)\ x_2(k)\ \ldots\ x_n(k) \right]^{\mathrm{T}}$ corresponds to the n transitions in the net that have both input and output places (i.e. those transitions which are neither sources nor sinks).

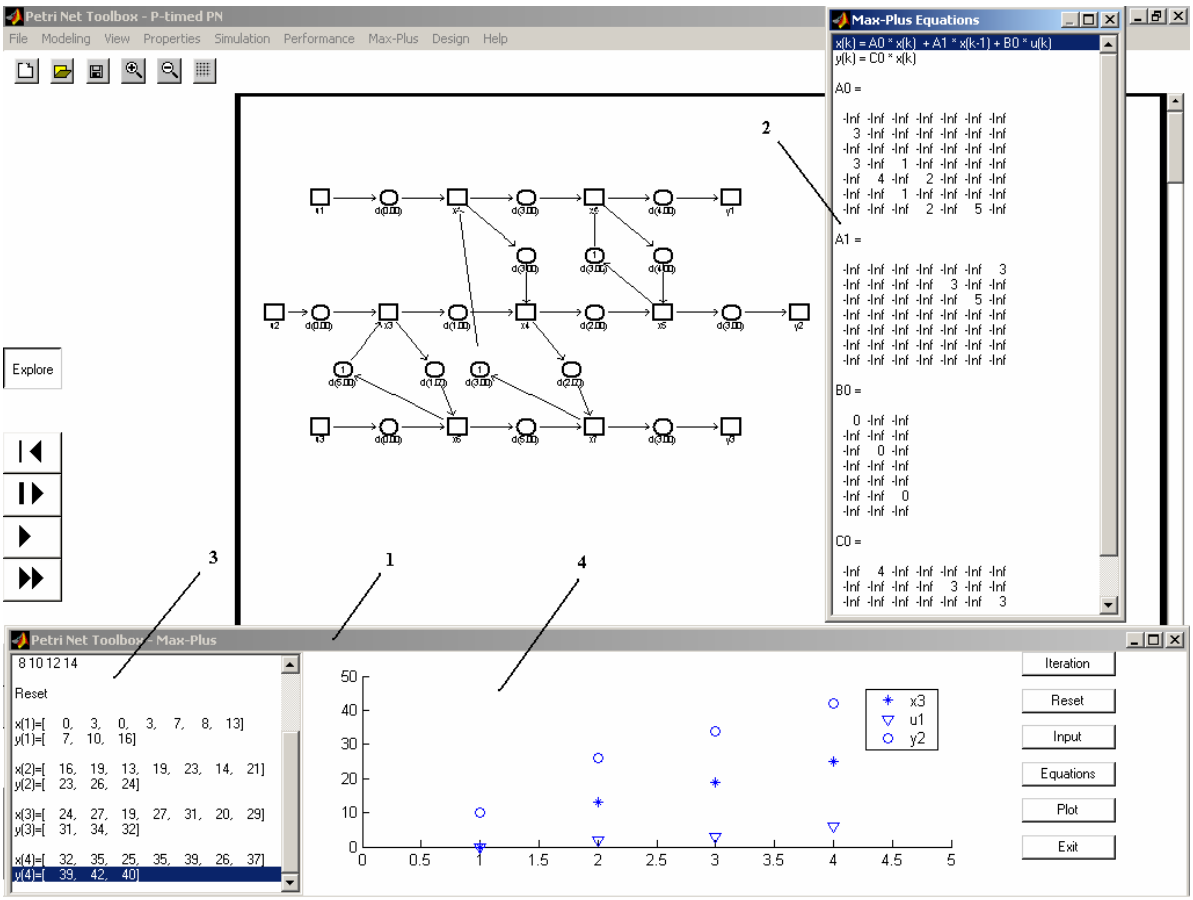


Fig. II.13. *Facilities available in the* ***PN Toolbox*** *for max-plus analysis.*

The max-plus model is automatically constructed by the ***PN Toolbox*** by using the ***Max-Plus*** menu. This command opens the ***Max-Plus*** window (denoted by (1) in fig. II.13) providing access to the following features available for max-plus analysis:

- ***Equations*** button: displays the max-plus equations in a separate window (denoted by (2) in fig. II.13);
- ***Input*** button: allows the user to set the values of the input vectors (time instants). This can be done (*i*) by introducing the time instants as a matrix with *m* rows and *N* columns directly in the *dialogue box*, or (*ii*) by creating the matrix ***input_time*** with the structure described above in MATLAB's *Workspace*;
- ***Plot***: allows the user to select the components of the input, state or output vectors that will be plotted, by using vectors with components equal to 0 (enabled) or 1 (disabled);
- ***Iteration***: runs an iteration of the simulation at a time. The values of the state and output vectors are displayed in the ***Message Box*** located on the left side of the ***Max-Plus*** window. The selected components are plotted;
- ***Reset***: resets the model to the initial state and clears the MATLAB axes;
- ***Exit***: closes the ***Max-Plus*** window and returns the control to the ***PN Toolbox*** GUI.

The ***Message Box*** (located in the left side of the ***Max-Plus*** window, denoted by (3) in fig. II.13) displays the input vectors as well as information reflecting the stage of the analysis (the state and output vectors after each iteration).

In the MATLAB axes (denoted by (4) in fig. II.13) placed in the middle of the ***Max-Plus*** window, all the selected components are plotted. The components of the input, state and output vectors are plotted with the symbols "∇", "$*$" and "$\circ$", respectively. For two or more components of the same vector, different colors are used with the same symbol.

II.6. Design

A facility for the synthesis of timed or (generalized) stochastic PN models is ***Design***, which allows exploring the dependence of a ***Design Index*** **(*I*)** on one or two ***Design Parameters*** that vary within intervals defined by the user.

A ***Design Parameter*** may be selected as (*i*) the initial marking of a place, (*ii*) a parameter of the distribution function defining the duration associated with a place or a transition in timed PNs, (*iii*) the mean value of the exponential distribution function associated with a transition in (generalized) stochastic PNs. The ***Design Parameters*** are generically denoted by ***x*** and ***y***; any other notation is not accepted. To ensure the correspondence between the symbol ***x*** (or ***y***) and the selected ***Design Parameter***, this symbol must be used as a numerical value when filling out the appropriate dialogue box (exactly as detailed in sections II.1.2 and II.1.3). The place or transition subject to parameterization is automatically colored in red.

The ***Design Index*** may be selected as a global performance index associated with a ***Design Node***, namely ***Service Rate***, ***Service Distance***, ***Service Time*** or ***Utilization*** for a transition, or ***Arrival Rate***, ***Arrival Distance***, ***Throughput Rate***, ***Throughput Distance***, ***Waiting Time*** or ***Queue Length*** for a place – see section II.4. The ***Design Index*** and the ***Design Parameter*** do not necessarily refer to the same node.

When selecting the ***Design*** command from the ***Menu Bar*** of the ***PN Toolbox***, a new window is opened (see fig. II.14). The choice of the ***Design Index*** and ***Design Node*** is made from the corresponding combo-boxes.

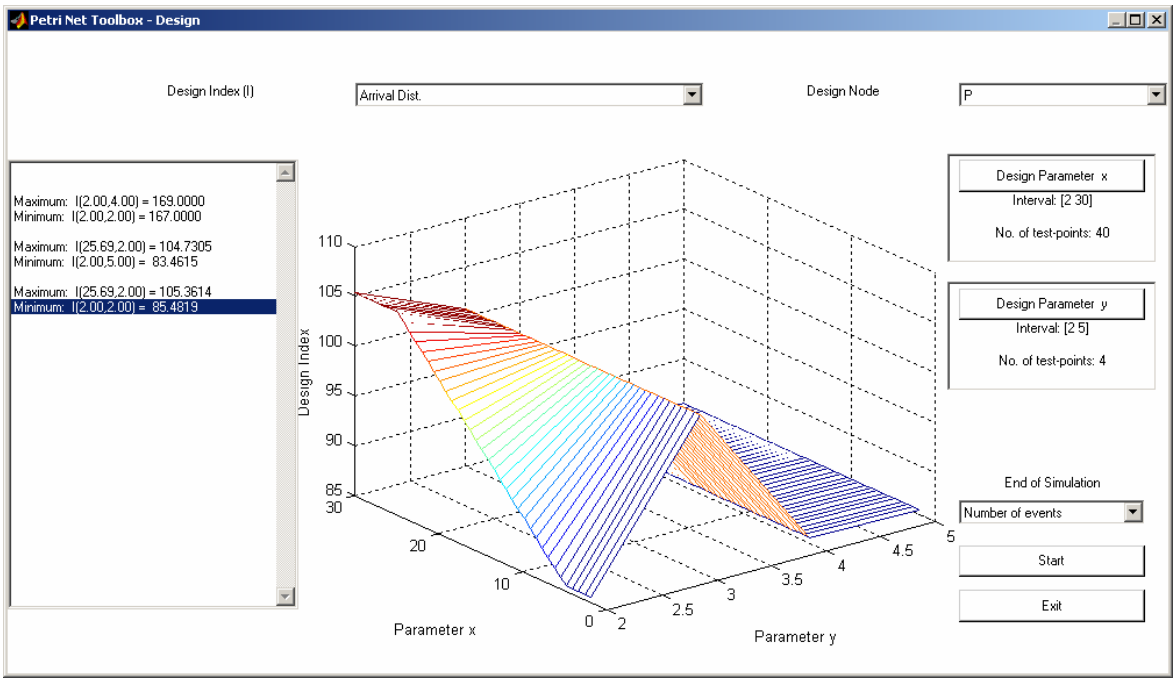
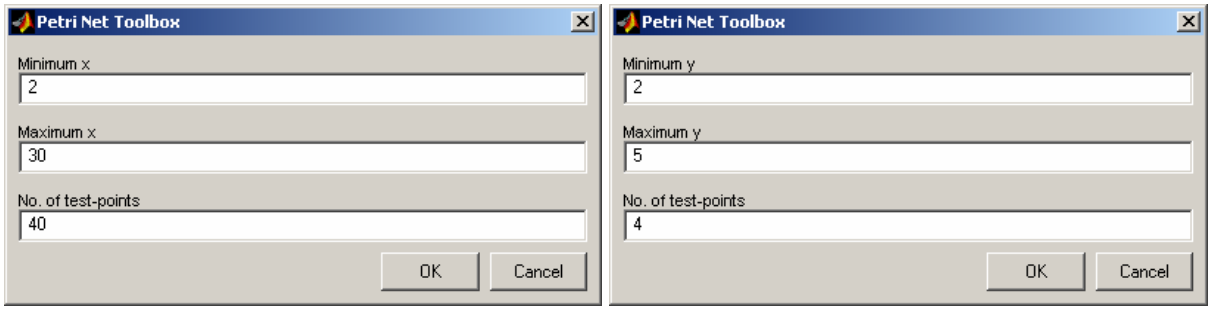


Fig. II.14. *The window opened by the* ***PN Toolbox*** *for* ***Design*** *option.*

By pushing the ***Design Parameter x*** button, the dialogue box presented in fig. II.15.(a) is opened, allowing the user to set the numerical information corresponding to ***x***, consisting in the extreme values to be considered and the number of equally-spaced test-points (where the ***Design Index*** will be calculated as commented bellow). The same way, numerical information can be set for ***y*** by pushing the ***Design Parameter y*** button, which opens the dialogue box presented in fig. II.15.(b).



(a) (b)

Fig. II.15. *The dialogue boxes for setting* ***Design Parameter*** (a) ***x*** *and* (b) ***y***.

The set of simulation-experiments defined by the ***Design Parameters*** is launched by pushing the ***Start*** button and cannot be interrupted until scanning all the test-points. The ***Exit*** button serves only for closing the ***Design*** window (i.e. after finishing all the simulation-experiments).

For each test-point (in the case of a single parameter) or for each pair of test-points (in the case of two parameters) automatically constructed by the ***PN Toolbox*** in accordance with the information given in the ***Design Parameter*** dialogue boxes, a simulation experiment is performed in the ***Run Fast*** mode. Each simulation experiment ends when the condition selected via the combo-box ***End of Simulation*** is fulfilled. The results of all these simulation experiments yield a graphical plot (2-D or 3-D, respectively) defining the dependence of the selected ***Design Index*** **(*I*)** on the ***Design Parameter*****(*s*)**. Besides the graphical plot, the toolbox displays the extreme values of the ***Design Index***, in the message box placed in the left side of the ***Design*** window.

Note that the simulation experiments performed for the given design parameter(s) provide the whole set of global indices associated with all the nodes of the net. Each of these indices can be visualized by appropriately choosing the ***Design Index*** and / or ***Design Node***. Such information remains available as long as the user does not press the buttons ***Design Parameter x***, ***Design Parameter y***, and does not alter the condition in the combo-box ***End of Simulation*** (because these operations reset the specific work variables).

Appendix 1: XML file-format of a file containing a PN model

The ***PN Toolbox*** uses an XML file for saving a model. This file is automatically generated by the ***PN Toolbox*** when the ***Save*** or ***Save As…*** commands are used. It can also be edited by using any text editor, if the format described below is respected.

The root tag for the PN model is **PNToolbox**. The tags for global information are:

- **Model_name** – the name of the file containing the model, including the extension ".xml";
- **Type** – a number that represents the PN type which can take one of the following values: 1 – untimed PN, 2 – T-timed PN, 3 – P-timed PN, 4 – stochastic PN, 5 – generalized stochastic PN;
- **Seed** – a positive integer used to initialize the random number generator of MATLAB;
- **Place** – the necessary information related to a place of the model, depending on the PN type;
- **Transition** – the necessary information related to a transition of the model, depending on the PN type;
- **Arc** – the information related to an arc of the model;
- **Probability** – the information related to the firing probabilities set for a group of conflicting transitions, if any;
- **Priority** – the information related to the firing priorities set for a group of conflicting transitions, if any.

The **Place** tag corresponding to a place in the PN model has the following structure:

- **Id** – the unique identifier of the place; a string in the form "px" with "x" a distinct positive integer;

- **Value** – the coordinates of the cell of the ***Drawing Area*** that contains the place (a couple of integers between 0 and 49);
- **Color** – a string representing the MATLAB color used for drawing the place in the ***Drawing Area***;
- **Label** – additional information used for the graphical representation of the place:
  - **Name** – label of the place that is shown in the ***Drawing Area*** (any string character);
  - **Offset** – a couple of real values representing the offset from the default position of the label (bottom of place);
  - **Visible** – boolean variable corresponding to the status of the label visibility ("yes" or "no").
- **InitialMarking** – initial marking of the place;
- **Capacity** – capacity of the place;
- **Time** – information corresponding to the probability distribution of the time duration assigned to the place; used only in the case of P-timed PNs:
  - **Distribution** – name of the MATLAB distribution function;
  - **Parameters** – values for the parameters of the selected distribution function.

The **Transition** tag corresponding to a transition in the PN model has the following structure:

- **Id** – the unique identifier of the transition; a string in the form "ty" with "y" a distinct positive integer;
- **Value** – the coordinates of the cell of the ***Drawing Area*** that contains the transition (a couple of integers between 0 and 49);
- **Color** – a string representing the MATLAB color used for drawing the transition in the ***Drawing Area***;
- **Message** – a string representing the message displayed by the ***PN Toolbox*** in the ***Message Box*** when the transition is fired;
- **Label** – additional information used for the graphical representation of the transition:
  - **Name** – label of the transition that is shown in the ***Drawing Area*** (any string character);
  - **Offset** – a couple of real values representing the offset from the default position of the label (bottom of transition);
  - **Visible** – boolean variable corresponding to the status of the label visibility ("yes" or "no").
- **Time** – information corresponding to the probability distribution of the time duration assigned to the transition; used only in the case of T-timed, stochastic and generalized stochastic PNs:
  - **Distribution** – name of the MATLAB distribution function;
  - **Parameters** – values for the parameters of the selected distribution function;
  - **Marking_Dependent** – boolean variable showing the dependence of the firing rate of the transition on the marking of its input places ("yes" or "no"); used only in the case of stochastic and generalized stochastic PNs.

The **Arc** tag corresponding to an arc in the PN model has the following structure:

- **Id** – the unique identifier of the arc; a string in the form "az" with "z" a distinct positive integer;
- **From** – the **Id** of the departure node;
- **To** – the **Id** of the arrival node;
- **Style** – arc style (1 – regular, 2 – bidirectional, 3 – inhibitor);
- **Type** – type of graphical representation of the arc (1 – line, 2 – cubic spline);
- **Cubic** – the coordinates in the ***Drawing Area*** of the third point defining the spline curve; used only for the arcs with type = 2;
- **Color** – a string representing the MATLAB color used for drawing the arc in the ***Drawing Area***;
- **Weight** – the weight of the arc.

The **Probability** tag has the following structure:

- **Transitions** – the Ids of the conflicting transitions (separated by comma) subject to the probability assignment;
- **Values** – the corresponding values of the firing probabilities (separated by comma); nonnegative values whose sum equals 1.

The **Priority** tag has the following structure:

- **Transitions** – the **Id**s of the conflicting transitions (separated by comma) subject to the priority assignment;
- **Values** – the corresponding values of the firing priorities (separated by comma); nonnegative values, 0 meaning the highest priority.

Example: The XML file corresponding to the PN model in fig. II.16 is given below.

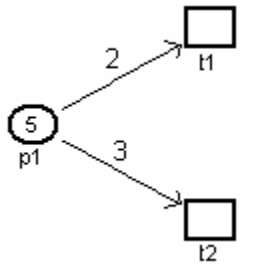


Fig. II.16. *ThePN model used for illustrating the XML file-format used by the **PN Toolbox**.*

```
<?xml version="1.0"?>
<PNToolbox>
        <Model_name>XMLexample.xml</Model_name>
        <Type>2</Type>  <!-- T-timed PN -->
        <Seed>66</Seed> <!-- initial seed -->
        <Place> <!-- place definition-->
                <Id>p1</Id> <!-- place's id -->
                <Value>5,43</Value> <!-- coordinates in the Drawing Area-->
                <Color>black</Color> <!-- drawing color-->
```

```
        <Label> <!-- information of the label-->
                <Name>p1</Name> <!-- string label-->
                <Offset>0.50,-0.20</Offset> <!-- the offset of the representation-->
                <Visible>yes</Visible> <!-- the visibility of the label-->
        </Label>
        <InitialMarking>5</InitialMarking> <!-- initial marking of the place-->
        <Capacity>Inf</Capacity> <!-- capacity of the place-->
</Place>
<Transition> <!—transition's definition-->
        <Id>t1</Id> <!-- transition's id-->
        <Value>8,45</Value> <!-- coordinates in the Drawing Area-->
        <Color>black</Color> <!-- drawing color-->
        <Message>Firing transition t1</Message> <!-- message displayed when it is fired-->
        <Label> <!-- information of the label-->
                <Name>t1</Name> <!-- string label-->
                <Offset>0.50,-0.20</Offset> <!-- the offset of the representation-->
                <Visible>yes</Visible> <!-- the visibility of the label-->
        </Label>
        <Time> <!-- information for time distribution function>
                <Distribution>constant</Distribution> <!-- constant distribution-->
                <Parameters>3</Parameters><!-- duration equal to 3-->
        </Time>
</Transition>
<Transition> <!-- second transition's definition -->
        <Id>t2</Id>
        <Value>8,41</Value>
        <Color>black</Color>
        <Message>Firing transition t2</Message>
        <Label>
                <Name>t2</Name>
                <Offset>0.50,-0.20</Offset>
                <Visible>yes</Visible>
        </Label>
        <Time>
                <Distribution>cont. uniform</Distribution> <!-- uniform distribution between 1 and 5-->
                <Parameters>1,5</Parameters>
        </Time>
</Transition>
<Arc> <!-- arc definition>
        <Id>a1</Id> <!-- arc's ID-->
        <From>p1</From> <!-- departure node-->
        <To>t1</To> <!-- arrival node-->
        <Style>1</Style> <!-- regular arc-->
```

```
            <Type>1</Type> <!—the graphical representation is a line -->
            <Color>black</Color> <!-- drawing color-->
            <Weight>2</Weight> <!-- arc's weight-->
        </Arc>
        <Arc>
            <Id>a2</Id>
            <From>p1</From>
            <To>t2</To>
            <Style>1</Style>
            <Type>1</Type>
            <Color>black</Color>
            <Weight>3</Weight>
        </Arc>
        <Probability> <!-- probability for conflicting transitions t1 and t2; 25% for t1 and 75% for t2 -->
            <Transitions>t1,t2</Transitions>
            <Values>0.25,0.75</Values>
        </Probability>
</PNToolbox>
```

Appendix 2: Configuration File for the Petri Net Toolbox

The ***PN Toolbox*** uses a configuration file to store the global information needed for initialization. This file, called *PNTconfig.txt*, is stored in the folder where the ***PN Toolbox*** is installed. The user can modify the values of all the parameters; the modifications are effective the next time when the ***PN Toolbox*** is started.

The options from *PNTconfig.txt* file used by ***PN Toolbox*** are:

- **Seed** – the initial state of the *random number generator* version 5 from MATLAB. This parameter is called "Seed" for history reasons; first version of ***PN Toolbox*** used version 4 of the *random number generator* where the 'seed' option of the MATLAB **rand** function was used. The version 5 uses 'state' parameters of **rand** function. (this parameter can have any integer value);
- **DisplayMessages** – boolean value that gives the status of the transitions' messages visibility in the ***Message Box*** in ***Step*** and ***Run Slow*** simulation modes (1 – the messages are visible, 0 – the messages are invisible);
- **Background** – the background color for all MATLAB figures opened by the ***PN Toolbox*** (a vector with three elements ranging from 0 to 1);
- **HTMLBackground** –the background color for all HTM files opened by the ***PN Toolbox*** (this parameter is given in hexadecimal);
- **HTMLText** – the text color for all HTML files opened by the ***PN Toolbox*** (given in hexadecimal);

- **Breakpoint** – the default breakpoint for ***Run Fast*** simulation mode (1 – Number of events, 2 – Simulation time, 3 – Firings for transition, 4 – Tokens arrived in place);
- **Breakpoint_Event** – default number of events, if breakpoint is selected on *Number of events*;
- **Breakpoint_Time** – default simulation time, if breakpoint is selected on *Simulation time*;
- **Breakpoint_Transition** – default index of the transition in the transitions array, if breakpoint is selected on *Firings for transition*;
- **Breakpoint_ServiceSum** – the number of firings for the **Breakpoint_Transition**;
- **Breakpoint_Place** – default index of the place in the places array, if breakpoint is selected on *Tokens arrived in place*;
- **Breakpoint_ArrivalSum** – the number of tokens arrived in the **Breakpoint_Place**;
- **Projects_Save** – the whole path of the folder where the ***PN Toolbox*** saves the new models;
- **Color_Wait_Token** – MATLAB color used for the graphical representation of places that contain reserved tokens (only in the case of *P*-timed PNs);
- **Color_Fire_Transition** – MATLAB color used for the graphical representation of firing transitions;
- **NewFileName** – the default name of the new model, including extension (.xml);
- **ColorPlaces** – MATLAB color used for drawing the places;
- **ColorTransitions** – MATLAB color used for drawing the transitions;
- **ColorArcs** – MATLAB color used for drawing the arcs.

Bibliografie

Ajmone Marsan, M., Balbo, G., Bobio, A., Chiola, G., Conte, G. and Cumani, A. (1985) "On Petri nets with stochastic timing", *Int. Workshop on Timed Petri Nets*, Torino, pp. 80-87.

Ajmone Marsan, M. (1990) "Stochastic Petri nets: An elementary introduction", in Rozenberg, G.: *Lecture Notes in Computer Science*, Vol. **424**, *Advances in Petri Nets 1989*, pp. 1-29, Springer-Verlag, Berlin.

Antsaklis, P.J. and Moody, J.O. (1998) *Supervisory Control of Discrete Event Systems Using Petri Nets*, Kluwer Academic Publishers.

ARENA Home Page: http://www.arenasimulation.com/support/.

Bolch, G., Greiner, S., de Meer, H. and Trivedi, K. (1998) *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*, John Wiley and Sons, New York.

Burns, A. and Wellings, A. (1991) *Real-Time Systems and Their Programming Languages*, Chap. 11, Addison–Wesley.

Cao, X.R. and Ho, Y.C. (1990) "Models of discrete event dynamic systems", *IEEE Control Systems Magazine*, Vol.**10**, pp. 69–75.

Bacelli, F., Cohen, G., Olsder, G.J. and Quadrat, J.P. (1992) *Synchronization and Linearity. An Algebra for Discrete Event Systems*, John Wiley and Sons, New York.

Cassandras, C.G. (1993) *Discrete Event Systems: Modeling and Performance Analysis*, Irwin.

Cassandras, C.G., Lafortune, S. and Olsder, G.J. (1995) "Introduction to the modelling, control and optimization of discrete event systems", in Isidori, A. (Ed.) *Trends in Control – A European Perspective*, Springer-Verlag, pp. 217–292.

Cohen, G., Dubois, D., Quadrat, J.P. and Viot, M. (1985) “A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing”. *IEEE Trans. Automat. Control*, Vol.**30**, pp. 210–220.

Cohen, G., Gaubert, S. and Quadrat, J.P. (1998) “Max-plus algebra and system theory: where we are and where to go now”, *IFAC Conf. on System Structure and Control*, Nantes.

Cohen, G., Gaubert, S. and Quadrat, J.P. (2001) “Max-plus algebra in Scilab and applications”, *Scilab Workshop*, Liama, Pekin.

David, R. et Alla, H. (1992) *Du Grafcet aux réseaux de Petri* (2e édition), Hermes, Paris.

David, R. and Alla, H. (1994) "Petri nets for modeling of dynamic systems – A survey", *Automatica*, Vol. **30**, pp. 175–202.

Desrochers, A.A. and Al-Jaar, R.Y. (1993) *Petri Nets for Automated Manufacturing Systems: Modeling, Control and Performance Analysis*, Rensselaer Polytechnic Institute.

Dijkstra, E. W. (1968) “Co-operating sequential processes”, in Genyus, F. (ed.) *Programming Languages*, New York Academic, pp. 43-112.

Freedman, P. (1991) "Time, Petri nets, and robotics", *IEEE Trans. on Robotics and Automation*, Vol. **7**, No. *4*, pp. 417-433.

Gürel, A., Pastravanu, O.C. and Lewis, F.L. (1994) "A robust approach in deadlock-free and live FMS design", *Proc. 2nd IEEE Mediterranean Symp. New Directions in Control and Automation*, pp. 40–47, Crete.

Gürel, A., Pastravanu, O.C. and Lewis, F.L. (1998) "A system-theoretic approach for discrete-event control of manufacturing systems", in Gunawardena, J. (Ed.) *Idempotency*, pp. 242-261, Cambridge University Press.

Ho, Y.C. and Cassandras, C.G. (1983) "A new approach to the analysis of discrete event dynamic systems", *Automatica*, Vol. **19**, pp. 149–167.

Huang, H.H., Lewis, F.L., Pastravanu, O.C. and Gürel, A. (1995) "Flowshop scheduling design in an FMS matrix framework", *Control Engineering Practice*, Vol. **3**, pp. 561–568.

Ichikawa, A. and Hiraishi, K. (1987) "Analysis and control of discrete event systems represented by Petri nets", *Lecture Notes in Control and Information Science*, Vol. **103**, pp. 115–134, Springer-Verlag.

Iordache, M.V. and Antsaklis, P.J. (2002) "Software tools for the supervisory control of Petri nets based on place invariants", Technical Report ISIS-2002-003, University of Notre Dame, IN, USA, http://www.nd.edu/~isis/techreports/isis-2002-003.pdf.

Jeng, M.D. and DiCesare, F. (1993) "A review of synthesis techniques for Petri nets with applications to automated manufacturing systems", *IEEE Trans. on Systems, Man and Cybernetics*, Vol. **23**, pp. 301–313.

Jeng, M.D. and DiCesare, F. (1995) "Synthesis using resource control nets for modelling shared-resource systems", *IEEE Trans. on Robotics and Automation*, Vol. **11**, pp. 317–327.

Jensen, K. (1992) *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*, *Vol. 1: Basic Concepts*, Springer-Verlag.

Jensen, K. (1994) *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*, *Vol. 2: Analysis Methods*, Springer-Verlag.

Jensen, K. (1997) *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*, *Vol. 3: Practical Use*, Springer-Verlag.

Jucan, T. şi Ţiplea, F.L. (1999) *Reţele Petri. Teorie şi practică*, Ed. Academiei Române.

Kelton, W.D., Sadowski, R.P. and Sadowski, D.A. (2002) *Simulation with Arena – 2nd ed*., McGraw Hill Companies, Inc.

Kleinrock, L. (1976) *Queueing Systems*, John Wiley and Sons, New York.

Krogh, B.H. and Beck, L. (1986) "Synthesis of place/transitions nets for simulation and control of manufacturing systems", *Proc. IFIP Symp. Large Scale Syst.,* Zurich, pp. 661–666.

Lafortune, S. and Yoo, H. (1990) "Some results on Petri net languages", *IEEE Trans. on Automatic Control*, Vol. **35**, pp. 482–485.

Leţia, T.S. şi Aştilean, A.M. (1998) *Sisteme cu evenimente discrete: modelare, analiză, sinteză şi control*, Ed. Albastră.

Levi. S.T. and Agrawala, A.K.(1990) *Real Time System Design*, Chapt. 7, McGraw–Hill.

Lewis, F.L. Pastravanu, O. and Huang, H.H. (1993) "Controler design and conflict resolution for discrete event manufacturing systems", *Proc. 32nd IEEE Conf. Decision and Control,* pp. 3288–3293, San Antonio.

Lewis, F.L., Huang, H.H., Pastravanu, O. and Gürel, A. (1995) "Control systems design for flexible manufacturing systems", in Raouf, A. and Ben-Daya, M. (Eds.) *Flexible Manufacturing Systems: Recent Developments*, pp. 253–290, Elsevier Science.

Lewis, F.L., Tacconi, D., Gürel, A., Huang, H.H. and Pastravanu, O. (1997) "Manufacturing controller design and deadlock avoidance using a matrix model for discrete event systems", in Tzafestas, S.G. (Ed.) *Methods and Applications of Intelligent Control*, pp. 485-508, Kluwer Academic Publishers.

Lewis, F.L., Gürel, A., Bogdan, S., Doganalp, A. and Pastravanu, O. (1998) "Analysis of deadlock and circular waits using a matrix model for flexible manufacturing systems", *Automatica*, Vol. **34**, No. 9, pp. 1083-1100.

Lindemann, C. (1998). *Performance Modelling with Deterministic and Stochastic Petri Nets*. John Wiley and Sons, Chichester.

Mahulea, C., Bârsan, L. and Pastravanu, O. (2001) "Matlab tools for Petri-net-based approaches to flexible manufacturing systems", In: Filip, F.G., Dumitrache, I. and Iliescu, S. (Eds.), *9th IFAC Symp. on Large Scale Systems LSS 2001*, Bucharest, pp. 184-189.

Mahulea, C. (2002) *Petri Net Toolbox – a MATLAB Library for Discrete Event Systems*, MS Thesis, Department of Automatic Control and Systems Engineering, The Sheffield University, UK.

Martinez, J. and Silva, M. (1982) "A simple and fast algorithm to obtain all invariants of a generalized Petri net", in Girault, C. and Reisig, W. (Eds.), *Application and Theory of Petri Nets*, Informatik Fachberichte 52, Springer, pp. 301-310.

Matcovschi, M.H., Mahulea, C. and Pastravanu, O. (2001) "Exploring structural properties of Petri nets in MATLAB", *7th Int. Symp. on Automatic Control and Computer Science SACCS 2001*, Iasi, CD Rom, 6 pg.

Matcovschi, M.H. and Mahulea, C.(2002) "Generalized stochastic Petri nets in performance evaluation for queueing networks", *ECIT'2002 and ROSYCS'2002 Joint Conf.*, Iasi, CD-ROM, 6 pg.

Matcovschi, M.H., Mahulea, C. and Pastravanu, O. (2002) "Computer tools for linear systems over max-plus algebra", *Periodica Politechnica*, "Politehnica" University of Timisoara, Romania, Trans. on Automatic Control and Computer Science, Vol. **47**(**61**), No. *1*, pp. 97-102.

Matcovschi, M.H., Mahulea, C. and Pastravanu, O. (2003) "Petri net toolbox for MATLAB", *11th IEEE Mediterranean Conference on Control and Automation MED'03*, Rhodes, Greece (submitted).

The MathWorks Inc., *Creating Graphical User Interfaces*, Natick, Massachusetts, 2000.

The MathWorks Inc., *Learning MATLAB*, Natick, Massachusetts, 2001.

The MathWorks Inc., *StateFlow User's Guide*, Natick, Massachusetts, 2001.

The MathWorks Inc. (2003) *MATLAB Connections - Third Party Products and Services*, http://www.mathworks.com/products/connections.

Mînzu, V. şi Cernega, D. (2001) *Sisteme dinamice cu evenimente discrete: abordări şi aplicaţii*, Ed. didactică şi pedagogică.

Mortensen, K.H. (2003) *Petri Nets Tools and Software*, http://www.daimi.au.dk/PetriNets/tools.

Murata, T. (1989) "Petri nets: properties, analysis and applications", *Proc. IEEE*, Vol. **77**, pp. 541–580.

Nissanke, N. (1997) *Real Time Systems*, Prentice Hall.

Pastravanu, O.C., Gürel, A., Huang, H.H. and Lewis, F.L. (1994) "Rule–based controller design algorithm for discrete event manufacturing systems", *Proc. American Control Conf.,* pp. 299–305, Baltimore.

Pastravanu, O.C., Gürel, A. and Lewis, F.L. (1994) "Petri net based deadlock analysis in flowshops with kanban-type controllers", *Proc. 10th ISPE/IFAC Int. Conf. on CAD/CAM, Robotics and Factories of Future*, Ottawa, pp. 75–80.

Pastravanu, O.C. (1997) *Sisteme cu evenimente discrete. Tehnici calitative bazate pe formalismul reţelelor Petri*, Ed. Matrix Rom.

Pastravanu, O.C., Gürel. A. and Lewis, F.L. (1997) "Teaching discrete event control of manufacturing systems", in Bir, A., Dinibutun, A.T., Eksin, I. and Istefanopulos,Y. (Ed.) *Proc. of the 4th IFAC Symposium on Advances in Control Education ACE'97*, Istanbul, pp. 307-312.

Petri, C.A (1962) *Kommunikation mit Automaten*, Institut für Instrumentelle Mathematik Bonn, Schriften des IIM Nr. 2.

Peterson, J.L. (1981) *Petri Net Theory and the Modeling of Systems*, Prentice–Hall.

Ramadge, P.J. and Wonham, W.M. (1987) "Supervisory control of a class of discrete event processes", *SIAM J. Control and Oprimization*, Vol. **25**, pp. 206–230.

Ramamoorthy, C.V. and Ho, G.S. (1980) "Performance evaluation of asynchronous concurrent systems using Petri nets", *IEEE Trans. on Software Eng.*, Vol. **6** (5), pp. 440-449.

Reisig, W. (1985) *Petri Nets – An Introduction*, Springer-Verlag.

Sanchez, A. (1996) *Formal Specification and Synthesis of Procedural Controllers for Process Systems*, *Lecture Notes in Control and Information Sciences*, Vol. **212**, Springer–Verlag.

Scilab Home Page, http://www-rocq.inria.fr/scilab/.

Stănescu, M., Caramihai, S.I., Curaj, A., Popescu, O., Popescu D.C., Catana, D. (1996) *Sisteme dinamice cu evenimente discrete*, Universitatea "Politehnica" Bucureşti (Tipar rotaprint).

Svádová, M. and Hanzálek, Z. (2001) "Matlab Toolbox for Petri Nets", *22nd Int. Conf. ICATPN 2001*, Newcastle, UK, pp. 32-36.

Yamalidou, E.C., Patsidou, E.P. and Kantor, J.C. (1990) "Modeling discrete-event dynamical systems for chemical process control – A survey of several new techniques", *Computers chem. Engng.*, Vol. **14**, No. *3*, pp. 281-299.

Zhou, M.C. and DiCesare, F. (1993) *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*, Kluwer, Boston.

Zurawski, R. and Zhou, M.C. (1994) "Petri nets and industrial applications: A tutorial", *IEEE Trans. on Ind. Electronics*, Vol. **41**, No. *6*, pp. 567-583.

Index şi dicţionar român-englez

A

accesibilitate <reachability> – 27
arbore de <reachability tree> – 29
graf de <reachability graph> – 30
mulţime de <reachability set> – 27
acoperire <coverability> – 29
arbore de <coverability tree> – 29
cu invarianţi – **v.** invariant
graf de <coverability graph> – 30
alegere <choice>
asimetrică <asymmetric choice> – 11
liberă <free choice> – 11
liberă extinsă <extended free choice> – 11
reţea cu – **v.** reţea
arbore <tree>
de accesibilitate – **v.** accesibilitate
de acoperire – **v.** acoperire
arc <arc> – 7
ponderea unui <weight of an arc> – 7
inhibitor <inhibitor arc> – 11

B

bloc corect format <well–behaved block> – 52
blocare circulară < circular blocking> – 29
buclă autonomă <self-loop> – 8

C

capacitate <capacity>
a unei poziţii – **v.** poziţie
a unei reţele – **v.** reţea
de jetoane – **v.** jeton
celulă <cell, workcell> – **v.** sistem (proces)
circuit orientat < directed circuit>– 33
concurenţă <concurrency> – 11
condiţie <condition> – 8
post-condiţie <post-condition> – 8
post-condiţie comună <common post-condition> – 11
pre-condiţie <pre-condition> – 8
conflict <conflict> – **v.** alegere
confuzie <confusion> – 11
asimetrică <asymmetric confusion> – 11
simetrică <symmetric confusion> – 11
conservativitate <conservativeness> – 86
consistenţă <consistency> – 87
controler (*procedural*) <(procedural) controller> – 66
criterii <criteria>
de accesibilitate – **v.** accesibilitate
de conservativitate – **v.** conservativitate
de consistenţă – **v.** consistenţă
de mărginire – **v.** mărginire
de repetitivitate – **v.** repetitivitate
de siguranţă – **v.** siguranţă
de viabilitate – **v.** viabilitate

D

deadlock – 29
decizie <decision> – **v.** alegere
distribuţie de probabilitate <probability distribution> – 129

E

ecuaţie de stare <state equation> – 30

eveniment <event> – 1, 8
excludere mutuală <mutual exclusion>
  generalizată <generalized mutual exclusion> – 64
  paralelă <parallel mutual exclusion> – 58
  secvenţială <sequential parallel exclusion> – 61
executarea unei tranziţii – **v.** tranziţie

F

fir de aşteptare <queue> – 171

G

graf <graph>
  de accesibilitate – **v.** accesibilitate
  de acoperire – **v.** acoperire
  marcat (*de evenimente*) <marked (event) graph> – 12
  tare conex <strongly connected graph> – 32

I

invariant <invariant> – 88
  de bază (*fundamental*) <basic invariant> – 89
  minimal <minimal invariant> – 89
  reţea acoperită cu <net covered by invariants> – 90
  suport de <invariant support> – 88

J

jeton <token> – 8
  capacitate de <token capacity> – 81
  rezervat <reserved token> – 103
job-shop – 125, 196

K

kanban – 62

L

lanţ Markov <Markov chain> – 155
  stare <state> – 137

M

marcaj <marking> – 7
  accesibil – **v.** accesibilitate
  de deadlock – **v.** deadlock
  iniţial <initial marking> – 8
  viabil – **v.** viabilitate
maşină de stare <state machine> – 12
matrice de incidenţă <incidence matrix> – 30
  de ieşire <output incidence matrix> – 31
  de intrare <input incidence matrix> – 31
max-plus <max-plus> – 175
  model de stare <state-space model> – 178
  operator Kleene <Kleene star> – 182
  operaţii <operations> – 175
mărginire <boundedness> – 27
  structurală <structural boundedness> – 85
metodă <method>
  poziţiilor complementare – **v.** poziţie
  rafinării operaţiilor – **v.** rafinare
model <model>
  calitativ, logic sau netemporizat <qualitative, logical or untimed model> – 10
  cantitativ sau temporizat <quantitative or timed model> – 105
modelare <modeling>
  a efectuării operaţiilor – **v.** operaţie
  a utilizării resurselor – **v.** resursă
  structuri tipice utilizate în <typical structures used in modeling> – 17
mulţime <set>
  de accesibilitate – **v.** accesibilitate
  predecesor <pre-set> – 8
  succesor <post-set> – 8

N

nod <node>
  al unei reţele Petri <node of a Petri net> – 7

O

operaţie
  rafinarea operaţiilor – **v.** rafinare
operator Kleene – **v.** max-plus

P

paralelism <parallel activities> – **v.** concurenţă
partajare <sharing>
a unei resurse – **v.** resursă
perioadă – **v.** regim periodic
pondere (a unui arc) – **v.** arc
post-condiţie – **v.** condiţie
poziţie <place> – 7
complementară (*metoda*) <complementary place> – 10
de capacitate finită <finite-capacity place> – 10
de capacitate infinită <infinite-capacity place> – 8
structural mărginită <structurally bounded place> – 85
temporizată <timed place > – 104
pre-condiţie – **v.** condiţie
proces stohastic <stochastic process>– 137
proiectare <design>
specificaţie de <design specification> - 49
unei structuri de conducere – **v.** structură de conducere
unui controler procedural – **v.** controler
proprietate <property>
comportamentală <behavioral property> – 27
structurală <structural property> – 85

R

rafinare <refinement>
operaţiilor <refinement of operations> – 50
regim <regime>
ciclic (*periodic*) <periodical operation> – 107, 109
permanent <steady state>– 157
regulă <rule>
tranziţiei – **v.** tranziţie
repetitivitate <repetitiveness> – 86
resursă <resource>
ataşarea resurselor <addition of resource places> – 55
de stocare <buffer> – 50, 56
excludere mutuală în utilizarea unei – **v.** excludere mutuală
generală <general resource> – 50
nepartajată <nonshared resource> – 50
partajată <shared resource> – 50
partajată paralel <parallelly shared resource> – 58, 63
partajată secvenţial <sequentially shared resource> – 60, 63
specifică <specific resource> – 50, 58
reţea (*Petri*) <(Petri) net>
acoperită cu invarianţi – **v.** invariant
cu alegeri asimetrice <asymmetric-choice net> – 13
cu alegeri libere <free-choice net> – 12
cu alegeri libere extinse <extended free-choice net> – 13
cu capacitate finită <finite-capacity net> – 10
cu capacitate infinită <infinite-capacity net> – 8
cu priorităţi < net with priorities > – 11
cu probabilităţi <net with probabilities> – 10
mărginită <bounded net> – **v.** mărginire
generalizată stohastică <generalized stochastic net> – 160
netemporizată <untimed (Petri) net> – 7
ordinară <ordinary (Petri) net> – 8, 11
pură <pure (Petri) net> – 8
reversibilă <reversible net> – **v.** reversibilitate
sigură <safe net> – 28
stohastică <stochastic net> – 153
tare conexă <strongly connected net> – **v.** graf tare conex
temporizată <timed (Petri) net> – 103
viabilă <live net> – **v.** viabilitate
reversibilitate <reversibility> – 28

S

secvenţă de executări (*de tranziţii*) <firing sequence> – 31
server <server> – 21, 138, 161, 163, 171

cu posibilități de defectare <unreliable server> – 22
sincronizare <synchronization> – 11
sinteză <synthesis> – 50
ascendentă <bottom-up synthesis> – 50
descendentă <top-down synthesis> – 50
hibridă <hybrid synthesis> – 50
sistem (*proces*) <system (process)>
de aşteptare <queueing system> – 171
de calcul <computer system> – 39, 67, 94, 100, 113, 119
de comunicație <communication system> – 18, 38, 111, 192
de fabricație (*automat*) <(automated) manufacturing system (cell)> – 25, 44, 70, 98, 118, 123, 142
de fabricație, flexibil <flexible manufacturing system> – 73
producător-consumator <producer-consummer system> – 17
specificație de proiectare - **v.** proiectare
stare <state>
a unei rețele Petri – **v.** marcaj
a unui lanț Markov – **v.** lanț Markov
a unui model max-plus – **v.** max-plus
structură de conducere <control structure> – 49
subclase de rețele Petri ordinare <subclasses of ordinary Petri nets> – 12
suport de invariant – **v.** invariant

T

temporizare <timing>
deterministă <deterministic timing > – 103
stohastică <stochastic timing > – 129
topologie (*de rețea Petri*) <(Petri net) topology> – 8
tranziție <transition> – 7
executarea unei <firing of a transition> – 9
prioritate în executare <firing priority> – 11
probabilitate de executare <firing probability> – 11
receptor <sink transition> – 9
regula (*simplă*) *a* <(weak) transition rule> – 9
regula strictă a <strict transition rule> – 10
sursă <source transition> – 9
temporizată <timed> – 103
validarea unei <enabling of a transition> – 9
validată de ordin q <*q*-enabled> – 153

V

validarea unei tranziții – **v.** tranziție
vector <vector>
de control <control vector> – 31
de marcaj – **v.** marcaj
de stare – **v.** stare
numărului de executări <firing count vector> – 32
viabilitate <liveness> – 28

APLICAȚII ALE REȚELELOR PETRI

Ideea de a elabora lucrarea de față s-a conturat pe fondul evoluției, într-un ritm deosebit de alert, a domeniului sistemelor cu evenimente discrete, din dorința de a oferi un material care să furnizeze atât informații teoretice ample, cât şi o bogată ilustrare a utilizării lor în practică. Deşi, prin conținutul transmis, cartea posedă un caracter monografic, prin modul de structurare vizează, drept principal obiectiv, construirea raționamentelor ştiințifice specifice domeniului, manipulând cunoştințele în contexte aplicative variate şi relevante. Fiecare capitol al cărții conține un substanțial breviar teoretic, oferind cititorului toate noțiunile şi rezultatele de bază care, ulterior, sunt angrenate în rezolvarea de probleme. O contribuție deosebită la dezvoltarea abilităților de investigare este adusă de o serie de aplicații ce compară şi corelează rezultatele rezolvărilor analitice şi, respectiv, computaționale.

În scopul abordării asistate de calculator a rezolvărilor a fost utilizat mediul Petri Net Toolbox, dezvoltat de autorii cărții spre a rula sub MATLAB – care se bucură de o largă popularitate în cercetarea academică. Întregul material a fost de aşa manieră conceput încât să asigure adresabilitate pentru o sferă largă de cititori cu formație tehnico-ştiințifică, incluzând specialişti, cercetători, cadre didactice, studenți şi doctoranzi.
Stilul de prezentare se concretizează într-o expunere autonomă, ce nu necesită alte lecturi în paralel, dar, totodată, cititorului preocupat de aprofundarea anumitor teme îi sunt indicate referințe bibliografice generoase. Modul de organizare al lucrării nu obligă la parcurgerea integrală a textului, permițând o lectură selectivă, orientată în conformitate cu nivelul de pregătire şi problematica de interes a fiecărui cititor în parte.

ÎN STUDIEREA SISTEMELOR CU EVENIMENTE DISCRETE

ISBN 973-8292-86-7